**MASTER EN SCIENCES MATHÉMATIQUES**

# Résolution approchée d'équations elliptiques au moyen de la méthode multigrille

LAMBOTTE, Marie-Agnès

*Award date:*
1984

*Awarding institution:*
Universite de Namur

# RESOLUTION APPROCHÉE D'ÉQUATIONS ELLIPTIQUES AU MOYEN DE LA MÉTHODE MULTIGRILLE.

MARIE_AGNES LAMBOTTE.

ANNÉE ACADÉMIQUE 83-84

PROMOTEUR : NGUYEN VAN HIEN

Il m'est agréable de présenter mes sincères remerciements à toutes les personnes qui m'ont permis de mener à bien la réalisation de ce mémoire

Je remercie tout spécialement :

Monsieur VERGISON, chargé du centre d'informatique scientifique de la S.A. SOLVAY, qui m'a apporté son expérience, son aide et ses encouragements au cours de ce travail ;

Monsieur le Professeur NGUYEN VAN HIEN pour la confiance qu'il m'a toujours accordée ;

Monsieur le Professeur WEXLER pour ses conseils judicieux.

M. Lambotte

# TABLE DES MATIERES

# 1. INTRODUCTION

Etant donné l'importance des modèles mathématiques gouvernés par des équations aux dérivées partielles en physique et en chimie (transferts thermiques, tranferts de masse, problèmes de potentiel, mécanique des fluides,...), l'étude des techniques de résolution de tels systèmes s'est beaucoup développée au cours de ces dernières années.

L'approche numérique couplée à la puissance des ordinateurs est la plus couramment utilisée. Elle conduit, dans le cas de problèmes linéaires ou linéarisés, à la résolution de grands systèmes creux c'est-à-dire comportant un grand nombre de composantes nulles.

La méthode multigrille à laquelle nous nous intéressons dans ce travail est un outil récent de résolution de tels systèmes qui connait un vif succès aujourd'hui.

Le chapitre 2 présente cette méthode en la situant dans le contexte des équations aux dérivées partielles sur un problème modèle : l'équation de Poisson.

Ce chapitre est divisé en trois parties :

- Dans la première, nous commençons par formuler analytiquement le problème à résoudre. En réalité, nous le présentons de trois manières différentes équivalentes donnant chacune un éclairage particulier sur la question : la formulation variationnelle, la formulation de Galerkin et la formulation différentielle classique. Cette dernière :

$$\begin{cases} \text{trouver } u \text{ suffisamment régulière solution de :} \\ \quad \begin{cases} -\Delta u(x,y) = f(x,y) \ , \ \forall (x,y) \in \Omega = (0,1)^2 \\ u(x,y) = g(x,y) \ , \ \forall (x,y) \in \Gamma = \partial\Omega \end{cases} \\ f \text{ et } g \text{ étant connues,} \end{cases}$$

a conduit à la méthode des différences finies. C'est de celle-ci dont il sera question dans la suite.

- Le paragraphe 2.2 est consacré à une présentation théorique de cette méthode numérique. Nous avons tenu à faire le lien entre le problème analytique et sa formulation discrète.

- La dernière partie de ce chapitre est consacrée à une analyse comparée de méthodes de résolution de systèmes linéaires en portant plus particulièrement notre attention sur les grands systèmes creux. Des programmes de calcul reprenant divers algorithmes de calcul sont repris en annexe.

Le chapitre 3 est entièrement consacré à la méthode multigrille :

- Dans le premier paragraphe, nous en décrivons les caractéristiques essentielles. Nous y abordons en particulier les problèmes de convergence et de l'effort de calcul que la méthode implique. Nous montrons ainsi qu'asymptotiquement, le nombre d'opérations requises est proportionnel au nombre de points de la grille de discrétisation ( N ) alors que les méthodes classiques en demandent $N^2$ ou N logN .

- Dans la deuxième partie, nous analysons plus en détail les propriétés de la méthode multigrille sur un exemple unidimensionnel. Nous y quantifions en particulier les différents paramètres qui la caractérisent tels le rayon spectral de la matrice d'itération et le facteur de proportionnalité donnant le nombre de calculs à effectuer.

- Dans la troisième partie, nous étendons de façon naturelle les résultats au cas bidimensionnel.

- Enfin, nous consacrons le dernier paragraphe aux applications numériques. Nous reprenons d'abord un résultat classique [1],[11] pour l'équation de Poisson pour l'étendre ensuite au cas de la résolution d'un problème non linéaire [12] .

Dans ce travail, nous avons surtout voulu mettre l'accent sur le fait que la méthode multigrille repose sur des bases saines et est bien adaptée pour résoudre des systèmes linéaires dérivant d'équations aux dérivées partielles.

Des essais comparatifs entre méthodes et sur des cas d'applications plus complexes devront le compléter. Mais dès à présent, en nous appuyant sur les résultats des essais effectués par Trottenberg et Stüben [10],[11], on peut se montrer optimistes quant à l'avenir de la méthode multigrille.

***

**  **

***

# 2. POSITION DU PROBLEME

## 2.1 FORMULATION FONCTIONNELLE DU PROBLEME

Beaucoup de problèmes de physique mathématique peuvent être décrits mathématiquement par une ou plusieurs équations fonctionnelles que nous écrivons sous la forme simplifiée :

$$L\,u = f$$

où $L$ opère d'un espace $X$ dans un espace $Y$

$f$ est donné dans $Y$

$u$ est cherché dans $X$

Dans ce travail, nous étudierons des modèles gouvernés par des équations aux dérivées partielles de type elliptique avec conditions aux limites de Dirichlet :

$$\begin{cases} L^{\Omega} u(x) = f^{\Omega} & x \in \Omega \\ L^{\Gamma} u(x) = f^{\Gamma} & x \in \Gamma := \partial\Omega \end{cases} \qquad (2.1.)$$

où $x = (x_1, \ldots, x_d) \in \mathbb{R}^d$

$\Omega \subset \mathbb{R}^d$ est un domaine ouvert borné de frontière $\Gamma$

$L^{\Omega}$ est un opérateur différentiel elliptique linéaire

$L^{\Gamma}$ est un opérateur de frontière linéaire

De tels problèmes peuvent parfois être formulés sous forme variationnelle :

$$\begin{cases} J(u) = \inf_{v \in V} J(v) \\ u \in V \end{cases} \qquad (2.2)$$

où $J(v) := \frac{1}{2} a(v,v) - f(v)$

et où $a(.,.)$ est une forme bilinéaire symétrique sur $V \times V$

$f(.)$ est une forme linéaire sur $V$

$V$ est une classe de fonctions représentant l'ensemble des états admissibles.

Afin de rendre l'exposé plus clair, nous analyserons le cas particulier de l'équation de Poisson sur le carré unité.

L'espace $V$ associé au problème est l'espace de SOBOLEV des fonctions de carré sommable dont les dérivées généralisées sont de carré sommable et nulles sur le bord du domaine $\Omega$ :

$$V = H_0^1(\Omega) = \left\{ v \in L^2(\Omega),\ \frac{\partial v}{\partial x_i} \in L^2(\Omega)\ ,\ i=1,2 ; v|_\Gamma = 0 \right\}$$

Le problème (2.1) se réécrit alors : Trouver $u \in H_0^1(\Omega)$ tel que

$$\begin{cases} L^\Omega u(x) := -\Delta u(x) = f^\Omega(x) & x \in \Omega = (0,1)^2 \\ L^\Gamma u(x) := \quad u(x) = 0 & x \in \Gamma \end{cases} \qquad (2.3)$$

c'est-à-dire où $L^\Omega$ est l'opérateur de Laplace

et $L^\Gamma$ est la restriction de la fonction u sur la frontière.

La formulation variationnelle correspondante s'écrit

$$\begin{cases} J(u) = \inf_{v \in H_0^1(\Omega)} J(v) \\ u \in H_0^1(\Omega) \end{cases} \qquad (2.4)$$

où $J(v) := \frac{1}{2} \iint_\Omega |\nabla v|^2 \, dx - \iint_\Omega f v \, dx$

On démontre alors la proposition suivante

Proposition 2.1.

| u est solution du problème (2.3) |
| --- |
| ssi |
| est solution du problème d'optimisation (2.4) |

La démonstration de cette proposition importante a été reprise en annexe.

Nous nous sommes limités à l'étude de l'équation de Poisson en montrant qu'elle dérivait d'un problème variationnel. Nous avons formalisé ce problème dans un cadre plus général (espaces fonctionnels). Des résultats utilisant ce formalisme ont été obtenus pour des systèmes d'équations elliptiques plus généraux [6]

Pour un même problème nous avons obtenu trois formulations équivalentes :

$$\begin{array}{l} \text{minimiser } J(v) \text{ sur } V \\ \Updownarrow \\ \text{résoudre } a(u,v) = L(v) \quad \forall v \in V \\ \Updownarrow \\ \text{résoudre } -\Delta u = f \end{array}$$

Nous allons reprendre ces différentes formulations et donner une idée des méthodes de résolution auxquelles elles conduisent.

### 2.1.2 Résoudre $a(u,v) = L(v) \quad \forall v \in V$

Comme dans le cas précédent, le problème est discrétisé sur un espace $V_n$ . Nous devons résoudre

$$a(u,v) = L(v) \qquad \forall v \in V_n$$

ou encore si $\{v_1, \ldots, v_n\}$ est une base de l'espace $V_n$

$$a(u,v_j) = L(v_j) \qquad \forall j = 1, \ldots, n$$

en remplaçant $u$ par $\sum_{i=1}^{n} \alpha_i v_i$ il vient

$$a\left(\sum_{i=1}^{n} \alpha_i v_i, v_j\right) = L(v_j) \qquad \forall j = 1, \ldots, n$$

de plus, $a$ étant bilinéaire, nous obtenons

$$\sum_{i=1}^{n} \alpha_i \, a(v_i, v_j) = L(v_j) \qquad \forall j = 1, \ldots, n$$

comme $a(.,.)$ et $L(.)$ sont connus, il s'agit de résoudre un système linéaire où les $\alpha_i$ , $i = 1, \ldots, n$ sont les inconnues .

Pour appliquer un tel schéma de résolution, il faudra résoudre des problèmes de :

- choix de l'espace $V_n$
- existence et unicité d'une solution $u_n$ du problème approché
- convergence de $u_n$ solution du problème approché vers $u$ solution du problème initial.

### 2.1.3 Résoudre $L\,u = f$

C'est par le biais de cette troisième formulation que nous allons aborder la résolution du problème.

Nous allons discrétiser l'équation $L\,u = f$

en $L_h\, u_h = f_h$ . Cette discrétisation ainsi que les problèmes qu'elle soulève sont étudiés en détail dans la suite.

Nous sommes maintenant en mesure de dresser un tableau mettant en parallèle formulations continues et formulations discrètes du problème :

| Problème continu | Problème discrétisé |
|---|---|
| - Min $J(v)$ , $v \in V$ | - Min $J^*(\alpha_1, \ldots, \alpha_n)$ , $\alpha_i \in \mathbb{R}$ |
| - Résoudre $a(u,v) = L(v)$ , $v \in V$ | - Résoudre le système linéaire $\sum_{i=1}^{n} \alpha_i (v_i, v_j) = L(v_j)$ , $j = 1, \ldots, n$ |
| - Résoudre $Lu = f$ | - Résoudre $L_h u_h = f_h$ |

Afin de présenter un travail complet, nous allons décrire le lien qui existe entre la formulation fonctionnelle de Galerkin $a(u,v) = L(v)$ et la formulation classique des différences finies. Cela fera l'objet du paragraphe suivant.

## 2.2 FORMULATION DISCRETE DU PROBLEME

On étudie l'équation de Poisson :

$$\begin{cases} -\Delta u = f & x \in \Omega = (0,1) \times (0,1) \subset \mathbb{R}^2 \\ u = 0 & x \in \partial\Omega = \Gamma \end{cases} \qquad (2.5.)$$

Posons $Q_h$ = le réseau de points qui maillent le carré fermé.

Soit $\sigma_M$ un parallélotope centré sur le point M et de côtés $(h_1, h_2)$

Soit $\sigma(M;1)$ la croix centrée sur M et de centre $\sigma$ (voir figure ci-dessous)

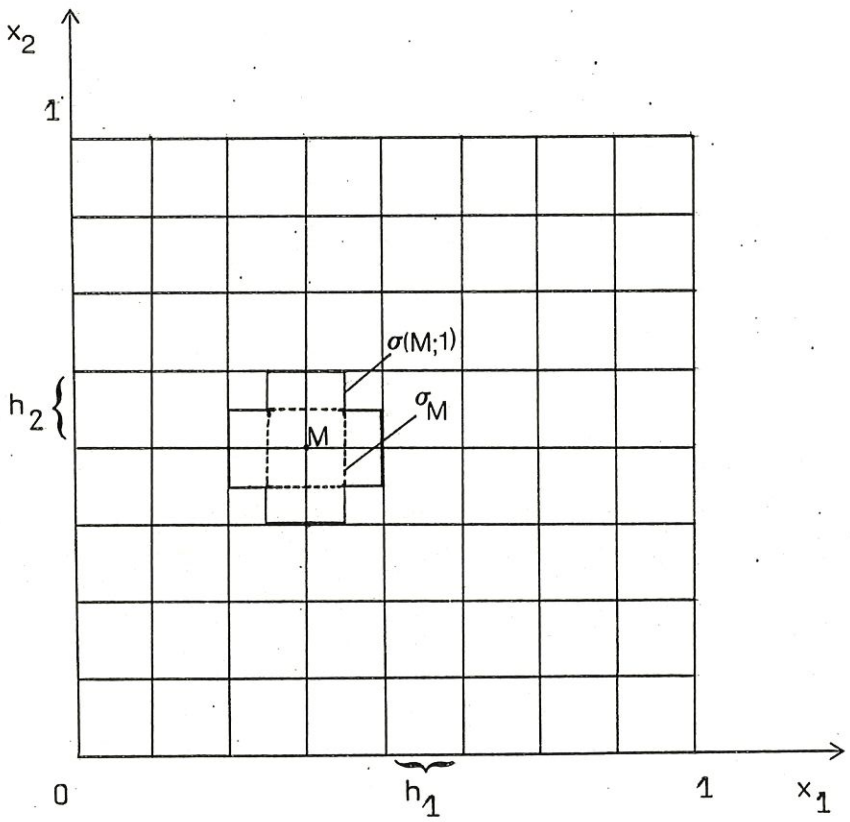



figure (2.1)

L'objectif de cette section est de répondre aux questions posées dans le paragraphe 2.1. c'est-à-dire définir concrètement l'espace $V_h$, montrer que la solution $u_h$ du problème approché existe et est unique et enfin, s'assurer de la convergence de la solution $u_h$ vers la solution $u$ du problème initial.

Nous avons à résoudre le problème analytique : trouver une fonction $u \in H_0^1((0,1)x(0,1))$ satisfaisant à :

$$a(u,v) = \sum_{i=1}^{2} \int_0^1 \int_0^1 \frac{\partial u}{\partial x_i} \frac{\partial v}{\partial x_i} \, dx_1 \, dx_2$$

$$= \int_0^1 \int_0^1 f.v \, dx_1 \, dx_2 = L(v) \qquad (2.6)$$

pour tout $v \in H_0^1((0,1)x(0,1))$

Discrétiser le problème revient à faire choix d'un espace finidimensionnel $V_h$ et d'une approximation $a_h(u,v)$ de la forme bilinéaire et $L_h(v)$ de la forme linéaire.

Définition de l'espace $V_h$

Soit $\Omega_h = \left\{ M \in Q_h \mid \sigma(M;1) \subset \Omega \right\}$

Soit $w_M$, la fonction caractéristique de $\sigma_M$ (remarquons toutefois que $w_M \notin H_0^1(\Omega)$ )

Définissons $V_h$ comme l'espace engendré par les fonctions $w_M$.

L'espace $V_h$ est une approximation externe de $H_0^1(\Omega)$ : $V_h \not\subset H_0^1(\Omega)$ et ce, contrairement à la méthode des éléments finis où on choisit $V_h \subset H_0^1(\Omega)$

Choix de la forme approchée de a : $a_h$

On approche la dérivée partielle $\frac{\partial}{\partial x_j}$ par un quotient différentiel $\delta_j$ :

$$\delta_j \varphi(x) = \frac{\varphi(x + h_j e_j/2) - \varphi(x - h_j e_j/2)}{h_j}$$

On définit alors $a_h$ comme

$$a_h(u,v) = \sum_{i=1}^{2} \int_0^1 \int_0^1 \delta_i u \; \delta_j v \; dx_1 \; dx_2$$

Le problème discret équivalent au problème (2.6) s'écrit :

$$\begin{cases} \text{Trouver une fonction } u_h \in V_h \text{ telle que :} \\ \\ a_h(u_h, v_h) = \int_0^1 \int_0^1 f_h \; v_h \; dx_1 \; dx_2 \\ \\ \text{pour tout } v_h \in V_h \end{cases}$$

Remarque : $f_h$ est une approximation de $f$ supposée continue.

Par exemple, en posant

$$f_h(M) = \frac{1}{h_1 h_2} \int_{M-\frac{h_1}{2}}^{M+\frac{h_1}{2}} \int_{M-\frac{h_2}{2}}^{M+\frac{h_2}{2}} f(x_1, x_2) \; dx_1 \; dx_2$$

$$= \frac{1}{h_1 h_2} \iint_{\sigma_M} f(x_1, x_2) \; dx_1 \; dx_2$$

il vient :

$$f_h(x) = \sum_M f_h(M) \; w_M(x)$$

En vertu des hypothèses que nous avons faites, en particulier sur $\Omega_h$ (on a choisi des croix donc on est loin du bord), on peut montrer aisément que la formule suivante d'intégration par partie est valable :

$$\int_0^1 \int_0^1 \sum_{i=1}^{2} \delta_i u_h \; \delta_i w_M \, dx_1 \, dx_2 = - \int_0^1 \int_0^1 \sum_{i=1}^{2} \delta_i^2 u_h \cdot w_M \; dx_1 \; dx_2$$

En effet :
$$\int_0^1\int_0^1 \sum_{i=1}^{2} \delta_1 u_h \, \delta_1 v_h \, dx_1 \, dx_2$$

$$= \int_0^1\int_0^1 \frac{u_h(x_1+\frac{h}{2},x_2)-u_h(x_1-\frac{h}{2},x_2)}{h} \cdot \frac{v_h(x_1+\frac{h}{2},x_2)-v_h(x_1-\frac{h}{2},x_2)}{h} dx_1 dx_2$$

$$= \frac{1}{h}\int_0^1\int_0^1 \frac{u_h(x_1,x_2)-u_h(x_1-h,x_2)}{h} v_h(x_1,x_2) \, dx_1 \, dx_2$$

$$- \frac{1}{h}\int_0^1\int_0^1 \frac{u_h(x_1+h,x_2)-u_h(x_1,x_2)}{h} v_h(x_1,x_2) \, dx_1 \, dx_2$$

$$= - \int_0^1\int_0^1 (\delta_1^2 u_h) \, v_h \, dx_1 \, dx_2$$

En procédant de façon analogue pour les dérivées en $x_2$ , il vient :

$$\int_0^1\int_0^1 (-\sum_{i=1}^{2} \delta_i^2 u_h) \, v_h \, dx_1 \, dx_2 = \int_0^1\int_0^1 f_h \, v_h \, dx_1 \, dx_2$$

pour tout $v_h \in V_h$

D'où en particulier pour tout $w_M \in V_h$ :

$$\int_0^1\int_0^1 (-\sum_{i=1}^{2} \delta_i^2 u_h) \, w_M \, dx_1 \, dx_2 = \int_0^1\int_0^1 f_h \, w_M \, dx_1 \, dx_2$$

ou encore :

$$-(\sum_{i=1}^{2} \delta_i^2 u_h(M)) \, h_1 \, h_2 = f_h(M) \, h_1 \, h_2$$

et l'on retrouve la formulation classique :

$$-\sum_{i=1}^{2} \delta_i^2 u_h(M) = f_h(M) \qquad M \in \Omega_h$$

ou encore si l'on réarrange les termes :

$$L_h \, u_h = f_h \qquad M \in \Omega_h$$

Revenons au problème approché écrit sous la forme :

$$a_h(u_h, w_h) = \int_0^1 \int_0^1 f_h \, w_h \, dx_1 \, dx_2$$

pour tout $w_M \in V_h$

La solution peut s'écrire :

$$u_h = \sum_{M \in \Omega_h} \alpha_M \, w_M$$

d'où

$$\begin{aligned} a_h(u_h, w_M) &= a_h\Big(\sum_{M' \in \Omega_h} \alpha_{M'} \, w_{M'}, w_M\Big) \\ &= \sum_{M' \in \Omega_h} \alpha_{M'} \, a_h(w_{M'}, w_M) \\ &= f_h(M) \end{aligned}$$

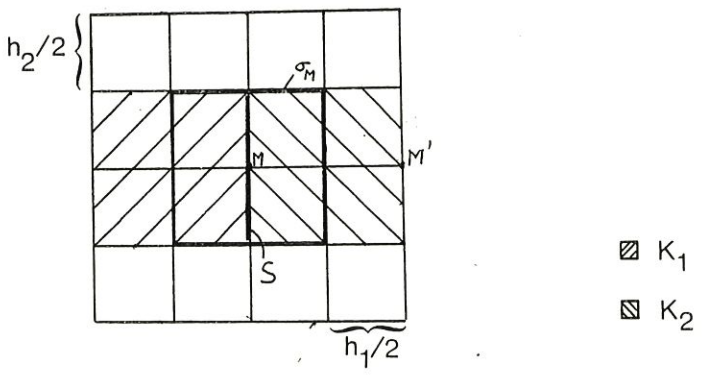
Calculons les éléments de la matrice de ce système linéaire :

Si $M = M'$ :

$$a_h(w_M, w_M) = \sum_{i=1}^{2} \int_0^1 \int_0^1 (\delta_i \, w_M)^2 \, dx_1 \, dx_2$$

Par définition de $\delta_j$ (cfr figure)

$$\delta_j w_M(x) = \begin{cases} 1/h_j & \text{sur } K_1 \\ 0 & \text{sur } S \\ -1/h_j & \text{sur } K_2 \\ 0 & \text{partout ailleurs} \end{cases}$$




figure(2.2)

D'où, pour $i = 1,2$ et $j \neq i$

$$\int_0^1\int_0^1 (\delta_i w_M)^2 \, dx_1 \, dx_2 = \frac{1}{h_i^2} \iint_{K_1 \cup K_2} dx_1 \, dx_2 = 2 \frac{h_j}{h_i}$$

et si $h_i = h_j$ on obtient :

$$a_h(w_M, w_M) = 4$$

Si $M \neq M'$ et si $M'$ est le centre d'un rectangle adjaçant à celui qui est centré sur $M$ (voir par exemple le cas de la figure (2.2) )

$$\int_0^1\int_0^1 \delta_i w_M \, \delta_i w_{M'} \, dx_1 \, dx_2 = - \frac{1}{h_i^2} \iint_{K_2} dx_1 \, dx_2$$

$$= - \frac{h_1 h_2}{h_i^2}$$

D'où, si $h_1 = h_2$

$$a_h(w_M, w_{M'}) = -1$$

Dans tous les autres cas :

$$\int_0^1\int_0^1 \delta_i w_M \, \delta_i w_{M'} \, dx_1 \, dx_2 = 0$$

Finalement, nous retrouvons encore la forme :

$$L_h u_h = f_h$$

En outre, on a la propriété de convergence :

Théorème 2.3. :

Quand $h \to 0$ : la suite $\{u_h\}$ converge vers $u$ dans $L^2(\Omega)$

$$u_h \xrightarrow{L^2(\Omega)} u$$

$$\delta_i u_h \xrightarrow{L^2(\Omega)} \frac{\partial u}{\partial x_i}$$

Nous ne démontrerons pas ce théorème qui fait appel à des éléments non triviaux d'analyse fonctionnelle. [6]

## 2.3. METHODES DE RESOLUTION NUMERIQUES D'UN SYSTEME LINEAIRE ASSOCIE A UN PROBLEME ELLIPTIQUE

La discrétisation de l'équation de Poisson nous a conduit à résoudre un système linéaire :

$$L_h \, u_h = f_h \quad .$$

de grande taille, à structure creuse c'est-à-dire possédant un grand nombre de zéros.

Dans l'arsenal des méthodes de résolution de systèmes linéaires, on distingue deux grandes familles :

- les méthodes directes et
- les méthodes itératives.

Ces dernières sont depuis longtemps très utilisées pour résoudre des systèmes dérivant de modèles mathématiques gouvernés par des équations aux dérivées partielles. Elles ont le mérite de prendre en compte la structure creuse des matrices et de lisser les erreurs d'arrondi.

Les premières qui étaient évitées à cause des problèmes de stabilité numérique qu'elles impliquent ainsi que du problème du temps calcul (la méthode de Gauss conduit à $\frac{n^3}{3}$ opérations, n désignant la dimension de la matrice) connaissent un regain d'intérêt depuis que des stratégies adaptées à ce type de systèmes linéaires ont été développées. Dans le paragraphe suivant, nous exposons certaines de ces méthodes, la suite du travail étant consacré à une méthode itérative particulière, la méthode multigrille.

## 2.3.1 METHODES DIRECTES

### Méthode de Gauss

Soit à résoudre le système linéaire :

$$A\,x = b \quad \text{où} \quad A \in \mathbb{R}^{n\times n}$$

$$x, b \in \mathbb{R}^{n\times 1}$$

L'idée de cette méthode est de décomposer la matrice $A$ en un produit de deux matrices l'une triangulaire inférieure ( $L$ ), l'autre triangulaire supérieure ( $U$ ) et de ramener ainsi la résolution du système :

$$A\,x = L\,U\,x = b$$

à celle de deux systèmes linéaires simples :

1) élimination gaussienne : on pose $U\,x = y$ et on résout le système triangulaire

$$L\,y = b$$

2) substitution inverse : ayant obtenu $y$ on résout ensuite le système :

$$U\,x = y$$

Ce qui revient à calculer la composante $x_i$ du vecteur $x$ à partir des composantes $x_{i+1}$ , ... , $x_n$.

Nous allons analyser plus en détail la procédure d'élimination gaussienne et montrer qu'elle se ramène simplement à prémultiplier les deux membres du système $A\,x = b$ par des matrices convenablement choisies.

A partir de l'élément $a_{11}$ de la matrice $A$, appelé pivot, on effectue les transformations élémentaires suivantes :

- remplacer la $k^{ième}$ ligne de $A$ (notée $A^k$) par

$$A^k - a_{k1}(a_{11})^{-1}A^1 \quad \text{pour } k \neq 1$$

c'est-à-dire effectuer l'opération

$$a_{k1}^k = a_{k1}^{k-1} - \frac{a_{k1}^{k-1}}{a_{11}^{k-1}} a_{11}^{k-1} \quad \text{pour } k \neq 1$$

ce qui revient à prémultiplier la matrice $A$ par une matrice de transformation triangulaire $M_1$.

Cette opération a pour effet de construire un matrice $M_1A$ dont les éléments sous diagonaux de la première colonne sont nuls

avec

$$M_1 = \begin{bmatrix} 1 & & & 0 \\ -m_{21} & \ddots & & \\ \vdots & & 1 & \\ & & & \ddots \\ -m_{n1} & & & 1 \end{bmatrix} \quad \text{et } m_{j1} = a_{j1}/a_{11} \quad \text{pour } 1 \leq j \leq n$$

- dans le même temps, on multiplie le second membre ( b ) par la matrice $M_1$.

On continue ainsi de proche en proche, ligne par ligne.

Ainsi, à la $i^{ième}$ ligne, à partir de l'élément $a_{ii}$ choisi comme pivot, on effectue les transformations élémentaires suivantes :

- remplacer $A^k$, la $k^{ième}$ ligne de A par

$$A^k - (a_{ii})^{-1} a_{ki} A^i \quad \text{pour } k < i$$

ce qui revient à prémultiplier la matrice des coéfficients $M_{i-1} \ldots M_1 A$ par une matrice de transformation $M_i$.

Les éléments sous-diagonaux de la $i^{ième}$ colonne de la matrice produit $M_i M_{i-1} \ldots M_1 A$ sont nuls

avec

$$M_i = \begin{bmatrix} 1 & & & \\ & \ddots & & \\ & & 1 & \\ & & -m_{i+1,i} & \ddots \\ & & \vdots & \\ & & -m_{ni} & 1 \end{bmatrix} \quad \text{et} \quad m_{ji} = a_{ji}/a_{ii} \quad \text{pour } i<j\leqslant n$$

- On effectue la même opération de prémultiplication sur le second membre ($M_{i-1} \ldots M_1 b$) du système.

Nous obtenons ainsi après la $n-1^{ième}$ étape :

$$M_{n-1} M_{n-2} \ldots M_2 M_1 Ax = M_{n-1} M_{n-2} \ldots M_2 M_1 b$$

ou, en posant $M_{n-1} \ldots M_2 M_1 = U$ et $M_{n-1} \ldots M_2 M_1 b = y$ ,

$$U x = y$$

où U est une matrice triangulaire supérieure.

La matrice $L^{-1} = M_{n-1} \ldots M_2 M_1$ a la forme :

$$L^{-1} = \begin{bmatrix} 1 & & & \\ -m_{21} & \ddots & & \\ \vdots & & 1 & \\ & & -m_{i+1,i} & \ddots \\ & & \vdots & \\ -m_{n1} & & -m_{ni} & 1 \end{bmatrix}$$

En procédant à ces différentes opérations de l'élimination gaussienne, nous avons en fait factorisé la matrice A en deux facteurs triangulaires :

$$A = L\,U$$

où L s'écrit :

$$L = \begin{bmatrix} 1 & & & \\ -m_{21} & \ddots & & \\ & 1 & & \\ & -m_{i+1\,i} & \ddots & \\ -m_{n1} & -m_{ni} & & 1 \end{bmatrix}$$

et U :

$$U = \begin{bmatrix} a_{11}^{(1)} & & a_{1i}^{(1)} & a_{1m}^{(1)} \\ & \ddots & & \\ & & a_{i-1\,i}^{(i)} & \\ & & a_{ii}^{(i)} & \\ & & & a_{m-1\,m}^{(i)} \\ & & & a_{mm}^{(i)} \end{bmatrix}$$

En toute généralité, le nombre d'opérations de multiplication ou de division nécessaires pour former L et U est de l'ordre de $n^3/3$ ce qui est beaucoup trop pour aborder l'étude de systèmes provenant d'équations aux dérivées partielles, ceux-ci étant souvent de grande taille et en outre creux, c'est-à-dire contenant un grand nombre de termes nuls.

En outre, dans ce qui précède, nous avons choisi comme pivots les éléments diagonaux des matrices obtenues dans la séquence d'élimination à commencer par l'élément $a_{11}$ de la matrice A pour continuer par l'élément $a_{22}$ de la matrice $M_1 A$ ,etc... Cette méthode de sélection de pivots présente toutefois de sérieux inconvénients. Considérons par exemple le système :

$$\begin{cases} x + 499\ y = 333 \\ x + \quad 1\ y = \quad 1 \end{cases}$$

Il a pour solution exacte :

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 1/3 \\ 2/3 \end{pmatrix}$$

Si nous effectuons un calcul approché à trois chiffres exacts, nous obtenons avec la méthode de Gauss par élimination simple:

$a_{11} = 1$ ,premier pivot

$$\left[\begin{array}{cc|c} 1 & 499 & 333 \\ 1 & 1 & 1 \end{array}\right]$$

avec

$$M_1 = \begin{pmatrix} 1 & 0 \\ -1 & 1 \end{pmatrix}$$

nous obtenons

$$\left[\begin{array}{cc|c} 1 & 499 & 333 \\ 1 & -498 & -332 \end{array}\right]$$

ce qui donne

$$y = -332/-498 \simeq 0,\ 67$$

$$x = 333 - 499.0{,}667 \simeq 0{,}167\ !$$

Il est donc indispensable si on veut tirer parti des méthodes directes pour résoudre des systèmes linéaires issus d'équations aux dérivées partielles :

1- de disposer d'une stratégie de choix de pivot robuste

2- d'éviter au maximum d'effectuer des opérations que la présence de termes nuls rend inutiles.

3- d'éviter de stocker de l'information inutile

Nous allons passer rapidement en revue les éléments principaux d'une stratégie possible (KEY : [5] ) pour résoudre un grand système creux (non nécessairement symétrique) par une méthode directe.

## CHOIX DU PIVOT

### 1- Gauss par pivotage complet.

Cette technique permet d'éviter les problèmes de précision de la méthode de Gauss par élimination simple. Elle consiste à choisir pour pivots des éléments suffisamment grands. Pratiquement, on cherche, parmi tous les éléments de la matrice, le plus grand élément n'ayant jamais appartenu à une ligne ou à une colonne pivot.
Reprenons l'exemple précédent mais en choisissant comme pivot le plus grand élément de la matrice A c'est-à-dire 499

$$\left[\begin{array}{cc|c} 1 & 499 & 333 \\ 1 & 1 & 1 \end{array}\right]$$

nous obtenons après pivotage

$$\left[\begin{array}{cc|c} 1 & 499 & 333 \\ 0.998 & 0 & 0.333 \end{array}\right]$$

d'où

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 0.333 \\ 0.667 \end{pmatrix}$$

Cette technique plus précise que la précédente implique toutefois un coût relativement élevé.

## 2- Gauss par pivotage partiel

Cette technique à mi-chemin entre les méthodes de Gauss par élimination simple et Gauss par pivotage complet permet d'éviter les erreurs dues à un manque de précision sans atteindre un coût trop élevé.
Dans ce procédé, la colonne de pivotage est choisie comme dans la méthode de Gauss par élimination simple. Le pivot étant choisi comme le plus grand coéfficient n'ayant jamais appartenu à une ligne pivot.
Il existe également des techniques de choix de pivot permettant de tirer un profit maximum des systèmes creux (KEY : [6])
En voici quelques unes.

## 3- Minimum ligne - Minimum colonne

Dans cette technique de sélection du pivot, la ligne pivot est celle comprenant le plus petit nombre d'éléments non nuls et n'ayant jamais été sélectionnée comme ligne pivot (dans le cas où plusieurs lignes satisfont cette condition, on choisit celle dont l'index de ligne est minimum).
La colonne pivot est déterminée en examinant toutes les colonnes correspondant aux éléments non nuls de la ligne pivot et en choisissant celle qui comprend un nombre minimum d'entrées (si plusieurs colonnes satisfont cette condition, on choisit celle dont l'index colonne est minimum).

4- Minimum colonne - Minimum ligne.

Cette technique est similaire à la précédente. On cherche d'abord la colonne comprenant le minimum d'entrées non nulles et qui n'a pas encore été sélectionnée. Ce sera la colonne pivot. Ensuite, les lignes contenant les éléments non nuls de la colonne pivot sont passées en revue pour déterminer celle comprenant le nombre minimum d'entrées.

5- Maximum colonne - Minimum ligne.

La colonne pivot est la colonne comprenant le maximum d'éléments non nuls et qui n'a pas encore été sélectionnée. La ligne pivot est déterminée de manière similaire au cas précédent.

6- Minimum du produit entre entrées colonne et entrées lignes.

Pour tout élément non nul de la matrice, on effectue le produit nombre d'entrées colonne non nulles x nombre d'entrées ligne non nulles. L'élément correspondant au produit minimum est choisi comme pivot. Si plusieurs éléments ont le même produit, l'élément correspondant à l'indice ligne minimum est sélectionné. Si ces éléments sont dans la même ligne, celui correspondant à l'indice colonne minimum est sélectionné.

A ces différentes possibilités de choix de pivot, on associe une stratégie de stockage très simple :

Soit par exemple la matrice

$$B = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 1 \\ 0 & 2 & 4 & 0 & 0 \\ 1 & 0 & 0 & 2 & 0 \\ 0 & 0 & 2 & 0 & 3 \end{bmatrix}$$

On lui associe deux matrices, respectivement A et ICOL contenant l'une les éléments non nuls de B et l'autre l'indice de la colonne correspondant aux différents éléments.

Il vient :

$$A = \begin{bmatrix} 3 & 0 \\ 2 & 1 \\ 2 & 4 \\ 1 & 2 \\ 2 & 3 \end{bmatrix} \quad \text{et} \quad ICOL = \begin{bmatrix} 1 & 0 \\ 2 & 5 \\ 2 & 3 \\ 1 & 4 \\ 3 & 5 \end{bmatrix}$$

Ces différentes stratégies (choix du pivot, stockage des éléments non nuls de la matrice du système) sont combinées dans un algorithme de Gauss programmé de manière à éviter les opérations inutiles.

Le code du programme de calcul est repris en annexe.

## 2.3.2. METHODES ITERATIVES

Ces méthodes de résolution d'un système d'équations linéaires ont été souvent associées à des systèmes provenant de la discrétisation d'équations aux dérivées partielles. C'est dans cette optique que nous les envisagerons ici.

Ces méthodes ont l'avantage de tenir compte de manière naturelle de la structure creuse du système et de lisser naturellement les erreurs d'arrondi bien que demandant en principe une infinité d'itérations pour obtenir la solution cherchée.

Parmi les plus connues et les plus simples, on peut citer celles de Jacobi et de Gauss-Seidel.

Soit $u_h^j$ , une approximation de la solution $u_h$ de $L_h u_h = f_h$ sur $\Omega_h$ .

Toutes les méthodes considérées dans ce chapitre peuvent s'exprimer sous la forme :

$$u_h^{j+1} = G\, u_h^j + c \quad , \quad j = 0,1,2,\ldots \qquad (2.7)$$

où G est la matrice d'itération $n \times n$ de la méthode et c est un vecteur associé connu.

La méthode est dite du premier degré dans la mesure où $u_h^{j+1}$ dépend explicitement seulement de $u_h^j$ et pas de $u_h^{j-1}, \ldots u_h^0$ .

Elle est linéaire puisque ni G , ni c ne dépendent de $u_h^j$ et stationnaire puisque ni G ,ni c ne dépendent de n .

Nous notons par :

$$v_h^j := u_h - u_h^j$$

l'erreur de l'approximation $u_h^j$ (aussi appelée correction de $u_h^j$ ), et par

$$d_h^j := f_h - L_h u_h^j$$

le défaut (ou résidu) de $u_h^j$ .

Puisque l'opérateur $L_h$ est linéaire, il est équivalent d'étudier le système :

$$L_h u_h = f_h$$

ou le système

$$L_h v_h^j = d_h^j \qquad (2.8)$$

Si dans (2.8) on remplace l'opérateur $L_h$ par un opérateur "plus simple" : $\hat{L}_h$ tel que $\hat{L}_h^{-1}$ existe,
la solution $\delta_h^j$ de :

$$\hat{L}_h \delta_h^j = d_h^j$$

fournit une nouvelle approximation $u_h^{j+1}$ de la solution $u_h$ par :

$$u_h^{j+1} = u_h^j + \delta_h^j$$

Si l'on se donne une valeur arbitraire $u_h^0$ de départ, ce procédé définit une procédure itérative dont

$$(I_h - \hat{L}_h^{-1} L_h) : \mathbb{G}(\Omega_h) \to \mathbb{G}(\Omega_h)$$

est la matrice d'itération ( $I_h$ représente l'identité sur $\Omega_h$ ).

En effet :

$$\begin{aligned} u_h^{j+1} &= u_h^j + \delta_h^j \\ &= u_h^j + \hat{L}_h^{-1} d_h^j \\ &= u_h^j + \hat{L}_h^{-1} (f_h - L_h u_h^j) \end{aligned}$$

d'où $u_h^{j+1} = (I_h - \hat{L}_h^{-1} L_h) u_h^j + \hat{L}_h^{-1} f_h \quad j=0,1,\ldots \qquad (2.9)$

et l'on retrouve la forme (2.7) avec $\mathbb{G} = (I_h - \hat{L}_h^{-1} L_h)$ et $c = \hat{L}_h^{-1} f_h$

Il nous reste à effectuer un choix judicieux de l'opérateur $\hat{L}_h$ .

Nous allons tout d'abord rappeler trois méthodes connues qui s'appliquent à des matrices générales : Jacobi, Jacobi relaxée et Gauss - Seidel.

## 1- Méthode de Jacobi

Dans la méthode de Jacobi, on remplace la matrice de l'opérateur $L_h$ par sa partie diagonale D (On décompose $L_h$ en $L_h = L + D + U$ où D , L , U sont des matrices composées respectivement des parties diagonale, triangulaire inférieure et triangulaire supérieure de la matrice de l'opérateur $L_h$ , tout le reste étant égal à zéro).

La relation (2.9) devient dans ce cas :

$$u_h^{j+1} = (I_h - D^{-1} L_h) u_h^j + D^{-1} f_h$$

$$j = 0,1,2,\cdots$$

ou encore plus simplement en fonction de la variable $v_h^j$ :

$$v_h^{j+1} = (I_h - D^{-1} L_h) v_h^j$$

$$j = 0,1,2,\cdots$$

Soulignons dès à présent les limitations de cette méthode.

Le rayon spectral de la matrice ( $I_h - D^{-1} L_h$ ) qui mesure la vitesse de convergence est égal à

$$\rho(I_h - D^{-1} L_h) = 1 - O(h^2)$$

Il est donc voisin de l'unité traduisant ainsi une convergence lente.

## 2- Méthode de Jacobi relaxée

Si nous introduisons un facteur de relaxation $\omega$ dans la méthode de Jacobi, les formules deviennent :

$$u_h^{j+1} = (I_h - \omega D^{-1} L_h) u_h^j + \omega D^{-1} f_h$$

$$j = 0,1,2,\cdots$$

et

$$v_h^{j+1} = (I_h - \omega D^{-1} L_h) v_h^j$$

$$j = 0,1,2,\cdots$$

Mêmes remarques que pour la méthode de Jacobi bien que la convergence soit plus rapide, elle reste lente.

Lorsque $0 < \omega \leqslant 1$ , il y a convergence :

$$\rho(I_h - \omega D^{-1} L_h) = 1 - 0(h^2)$$

Si par contre $\omega \leqslant 0$ ou $\omega > 1$ :

$\rho(I_h - \omega D^{-1} L_h)$ (si $h$ est suffisamment petit),

la procédure diverge.

## 3- Méthode de Gauss-Seidel

La matrice de l'opérateur $L_h$ est remplacée ici par ses parties triangulaire inférieure et diagonale.

La formule (2.9) nous donne cette fois :

$$u_h^{j+1} = (I_h - (L+D)^{-1} L_h) u_h^j + (L+D)^{-1} f_h$$
$$j = 0,1,2,\ldots$$

et

$$v_h^{j+1} = (I_h - (L+D)^{-1} L_h) v_h^j$$
$$j = 0,1,2,\ldots$$

Le rayon spectral de la matrice d'itération est ici de l'ordre de $1-0(h)$. La convergence bien que plus rapide que dans la procédure de Jacobi n'est pas exceptionnelle.

Dans la suite, ces trois premières méthodes seront appelées méthodes de relaxation. Remarquons que bien qu'elles aient été appliquées à un système dérivant d'une équation aux dérivées partielles elles sont en réalité très générales. La méthode suivante est par contre plus liée au problème discret sous-jacent.

## 4- Méthode de correction par passage à une grille grossière.

On peut aussi utiliser pour $\hat{L}_h$ , une approximation $L_H$ de $L_h$ sur une grille plus grossière $\Omega_H$ , ceci dans le but de réduire le nombre d'opérations à effectuer.

Ce qui signifie que la résolution du système (2.8) :

$$L_h \, v_h^j = d_h^j \qquad , \quad j=0,1,2,\ldots$$

sera remplacée par la résolution de :

$$L_H \, \delta_H^j = d_H^j \qquad , j = 0,1,2,\ldots \tag{2.10}$$

où $L_H : \mathbb{G}(\Omega_H) \to \mathbb{G}(\Omega_H)$ avec dim $\mathbb{G}(\Omega_H) <$ dim $\mathbb{G}(\Omega_H)$ et où l'on suppose que $L_H^{-1}$ existe.

Comme $\delta_H^j$ et $d_H^j$ sont des fonctions de grille sur $\Omega_H$ , deux opérateurs de transfert sont nécessaires :

1) Un opérateur de restriction : $I_h^H$

$I_h^H : \mathbb{G}(\Omega_h) \to \mathbb{G}(\Omega_H)$

envoie le défaut $d_h^j$ (donné sur $\Omega_h$ ) sur sa restriction $d_H^j$ (sur $\Omega_H$ )

$$d_H^j := I_h^H \, d_h^j$$

2) Un opérateur d'interpolation : $I_H^h$

$I_H^h : \mathbb{G}(\Omega_H) \to \mathbb{G}(\Omega_h)$

envoie l'erreur $\delta_H^j$ (calculée sur $\Omega_H$ ) sur sa prolongée, $\delta_h^j$ (sur $\Omega_h$ )

$$\delta_h^j := I_H^h \, \delta_H^j$$

Une itération de cette méthode se décompose comme suit :

1) Calcul du défaut $d_h^j := f_h - L_h u_h^j$

2) Restriction du défaut $d_H^j := I_h^H d_h^j$

3) Résolution exacte sur $\Omega_H$ de : $L_H \delta_H^j = d_H^j$

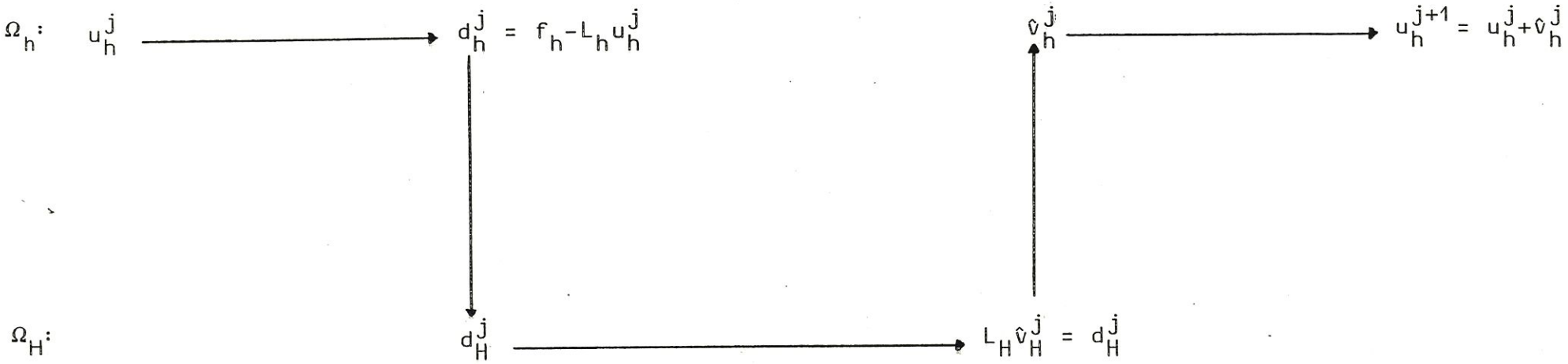
qui donne la correction $\delta_H^j$

4) Interpolation de la correction $\delta_h^j := I_H^h \delta_H^j$

5) Calcul de la nouvelle approximation $u_h^{j+1} := u_h^j + \delta_h^j$

La figure suivante schématise ces différentes opérations

Structure du procédé de correction par grille grossière.




La matrice d'itération associée est donnée par :

$$(I_h - I_H^h L_H^{-1} I_h^H L_h)$$

Par la formule (2.9) on a :

$$u_h^{j+1} = (I_h - I_H^h L_H^{-1} I_h^H L_h) u_h^j + I_H^h L_H^{-1} I_h^H f_h$$

$$j = 0,1,2,\ldots$$

et

$$v_h^{j+1} = (I_h - I_H^h L_H^{-1} I_h^H L_h) v_h^j$$

$$j = 0,1,2,\ldots$$

Soulignons que cette procédure utilisée seule ( $I_h^H$ , $I_H^h$ opérateurs linéaires) n'est pas convergente c'est-à-dire l'équation des résidus (2.10) sur la grille grossière $\Omega_H$ n'est pas une bonne approximation de l'équation originale des résidus (2.8) sur $\Omega_h$ . Le procédé est même divergent. On montre en effet aisémment que le rayon spectral de la matrice d'itération associée à la méthode de correction par grille grossière est plus grand ou égal à un

$$\rho(I_h - I_H^h L_H^{-1} I_h^H L_h) \geq 1 \qquad (2.11)$$

Il suffit de prendre un vecteur $w_h$ non nul dans $\mathbb{G}(\Omega_h)$ tel que : $L_h w_h$ appartienne au noyau de $I_h^H$ .
Un tel $w_h$ sera exactement reproduit par la matrice d'itération.
D'où (2.11).

Les méthodes qui viennent d'être décrites sont simples dans leur conception mais présentent de nombreux inconvénients : convergence lente voire divergence. Nous allons montrer dans le chapitre suivant qu'en les combinant astucieusement on peut obtenir une méthode de résolution très performante.

# 3. METHODE MULTIGRILLE

## 3.1 GENERALITES

L'idée de base de cette méthode est de combiner les méthodes de relaxation et de correction par grille grossière (décrites prédédemment) en utilisant les avantages de chacune d'elles.

Notons $\bar{w}_h = RELAX^{\nu}(w_h, L_h, f_h)$

où $\bar{w}_h$ est le résultat de $\nu$ pas de relaxation appliqués à l'équation $L_h u_h = f_h$ avec $w_h$ pris comme première approximation.

Description d'une itération deux-grilles

Partant de l'approximation $u_h^j$, on obtiendra $u_h^{j+1}$ en effectuant les opérations suivantes :

(1) Phase de lissage

- Calcul de $\bar{u}_h^j$ par l'application à $u_h^j$ de $\nu_1$ ($\geqslant 0$) pas d'une méthode de relaxation donnée.

$$\bar{u}_h^j := RELAX^{\nu_1}(u_h^j, L_h, f_h)$$

(2) Phase de correction par passage à une grille grossière (C G C)

- Calcul du défaut : $\bar{d}_h^j := f_h - L_h \bar{u}_h^j$
- Restriction du défaut : $\bar{d}_H^j := I_h^H \bar{d}_h^j$
- Résolution exacte sur $\Omega_H$ de : $L_H \hat{v}_H^j = \bar{d}_H^j$
  qui donne la correction : $\hat{v}_H^j$
- Interpolation de la correction : $\hat{v}_h^j := I_H^h \hat{v}_H^j$
- Calcul de l'approximation corrigée : $\bar{u}_h^j + \hat{v}_h^j$

(3) Phase de lissage

- Calcul de $\bar{u}_h^{j+1}$ , nouvelle approximation, par l'application à $\bar{u}_h^j + \hat{v}_h^j$ de $\nu_2$ ($\geqslant 0$) pas de la méthode de relaxation donnée.

$$u_h^{j+1} := \text{RELAX}^{\nu_2}( \bar{u}_h^j + \hat{v}_h^j , L_h , f_h )$$

La matrice d'itération de la méthode deux-grilles $(h,H)$ décrite ci-dessus est donnée par :

$$M_h^H = S_h^{\nu_2} K_h^H S_h^{\nu_1}$$

avec

$$K_h^H = I_h - I_H^h L_H^{-1} I_h^H L_h$$

où - $S_h$ représente la matrice d'itération de la procédure de relaxation choisie

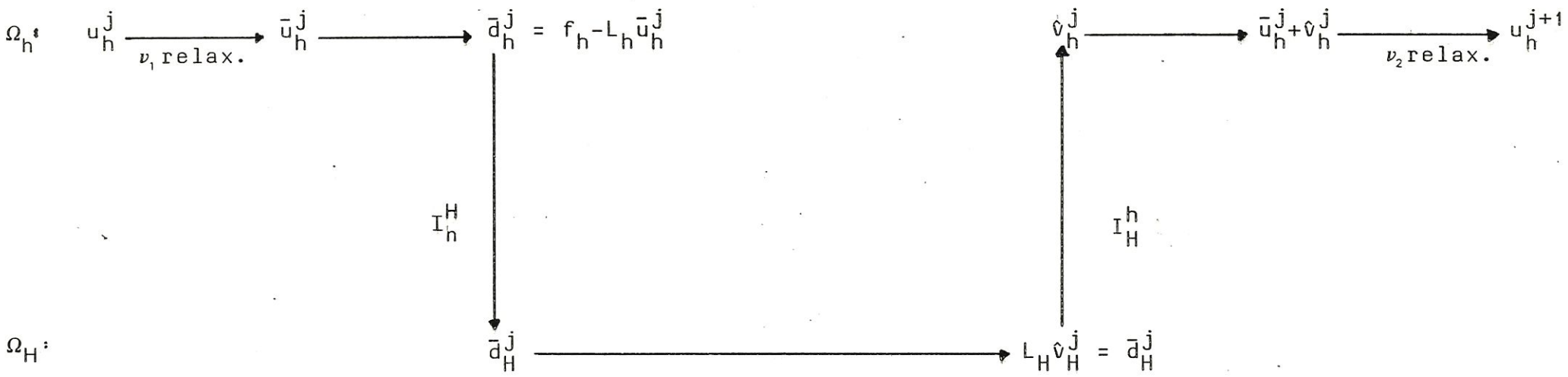
- $K_h^H$ représente la matrice d'itération associée à la méthode C.G.C.

Différentes composantes de la méthode deux-grilles seront à spécifier :

- le lissage (caractérisé par $S_h$)
- les nombres $\nu_1$ et $\nu_2$
- la grille grossière $\Omega_H$
- l'opérateur de restriction $I_h^H$
- l'opérateur d'interpolation $I_H^h$
- l'opérateur $L_H$ sur la grille grossière

La figure ci-dessous schématise les différentes opérations

## STRUCTURE D'UNE ITERATION DEUX-GRILLES




LA MATRICE D'ITERATION ASSOCIEE EST DONNEE PAR :

$$(S_h^{\nu_2} (I_h - I_H^h L_H^{-1} I_h^H L_h) S_h^{\nu_1})$$

Description d'une itération multigrille

Dans la section précédente, nous avons décrit le principe multigrille seulement dans sa version deux-grilles. Nous allons décrire à présent la méthode multigrille générale.

Rappelons les différentes étapes d'une itération deux-grilles:

- lissage
- restriction du défaut sur la grille grossière
- résolution exacte sur la grille grossière de l'équation du résidu : $L_H \hat{v}_H^j = \bar{d}_H^j$
- interpolation de la correction sur la grille fine
- lissage

Observons qu'il n'est pas nécessaire de résoudre exactement l'équation du défaut sur la grille grossière. En considérant $\Omega_H$ comme la grille fine et en introduisant une grille plus grossière que $\Omega_H$, nous pouvons remplacer la résolution exacte sur la grille $\Omega_H$ par une seconde application de la méthode deux-grilles et ainsi de suite.
Cette idée peut être appliquée récurssivement en utilisant des grilles de plus en plus grossières. Une méthode de résolution exacte est alors appliquée sur la grille la plus grossière.

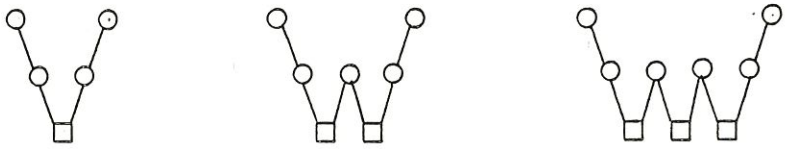
Voici quelques schémas illustrant la structure d'un pas d'itération d'une méthode multigrille. Les symboles O , □ , \ et / signifient respectivement lissage, résolution exacte, restriction et interpolation. Le paramètre $\gamma$ représente le nombre d'itérations à un niveau donné.

Structure d'une itération multigrille pour différents nombres de grilles et différentes valeurs de $\gamma$

* méthode deux-grilles

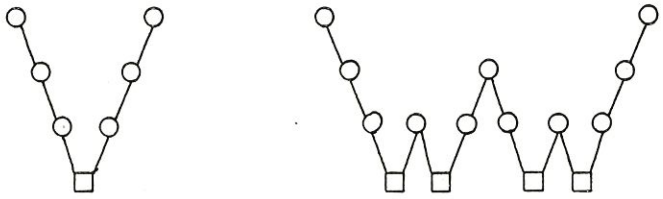
$\gamma = 1$

* méthode trois-grilles



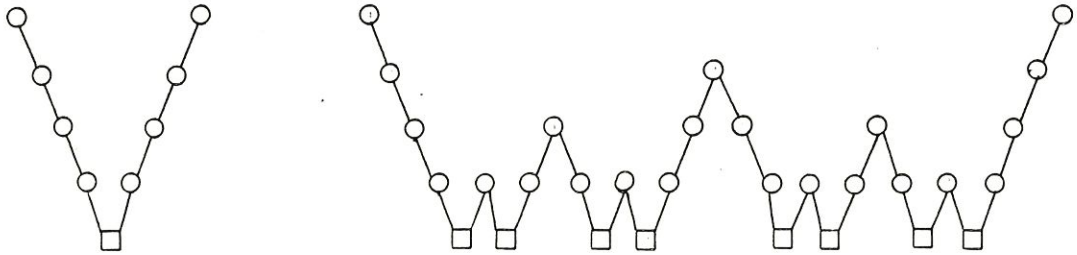
$\gamma = 1$ $\gamma = 2$ $\gamma = 3$

* méthode quatre-grilles



$\gamma = 1$ $\gamma = 2$

* méthode cinq-grilles



$\gamma = 1$ $\gamma = 2$

Afin de décrire formellement la méthode multigrille, nous aurons besoin d'une suite de grilles de plus en plus fines $\Omega_{h_l}$ caractérisée par leur pas de discrétisation $h_l (l=0,1,2\ldots)$ respectifs.

Pour simplifier les notations, nous remplacerons dans la suite l'index $h_l$ par $l$ (pour les grilles, fonctions de grilles et opérateurs de grilles).

Pour chaque grille $\Omega_h$ , nous supposons que les opérateurs linéaires :

$$L_l : \mathbb{G}(\Omega_l) \to \mathbb{G}(\Omega_l)$$
$$S_l : \mathbb{G}(\Omega_l) \to \mathbb{G}(\Omega_l)$$
$$I_l^{l-1} : \mathbb{G}(\Omega_l) \to \mathbb{G}(\Omega_{l-1})$$
$$I_{l-1}^{l} : \mathbb{G}(\Omega_{l-1}) \to \mathbb{G}(\Omega_l)$$

et les équations

$$L_l\, u_l = f_l \quad \text{sur} \quad \Omega_l$$

sont donnés.

Où $L_l$ est un opérateur inversible, $\mathbb{G}(\Omega_l)$ représente l'espace des fonctions de grille sur $\Omega_l$ et $S_l$ est l'opérateur linéaire d'itération correspondant à la méthode de relaxation donnée.

Le résultat de $\nu$ pas de relaxation appliqués à $L_l u_l = f_l$ avec $w_l$ pour première approximation est noté :

$$\bar{w}_l = \mathrm{RELAX}^{\nu}(w_l, L_l, f_l)$$

Voici la description d'un pas d'une itération multigrille - plus précisémment d'une itération $(l+1)$-grilles- pour résoudre l'équation :

$$L_l\, u_l = f_l \quad \text{sur} \quad \Omega_l$$

avec $l$ fixé $\geqslant 1$

Nous utilisons les grilles $\Omega_k$ et les opérateurs $L_k$ , $S_k$ $I_k^{k-1}$ , $I_{k-1}^{k}$ , $(k=l, l-1, l-2, \ldots, 0)$ . Nous supposons les paramètres $\nu_1$ , $\nu_2$ et $\gamma$ fixés.

Partant de l'approximation $u_l^j$ de $u_l$ , nous obtenons la nouvelle approximation $u_l^{j+1}$ en effectuant les opérations suivantes :

Si $l = 1$ : procéder comme lors de la méthode deux-grilles en notant $\Omega_h$ et $\Omega_H$ respectivement $\Omega_1$ et $\Omega_0$

Si $l > 1$ :

(1) Phase de lissage

- Calcul de $\bar{u}_l^j$ en appliquant à $u_l^j$ $\nu_1$ $(\geqslant 0)$ pas d'une méthode de relaxation donnée

$$\bar{u}_l^j := \mathrm{RELAX}^{\nu_1}(u_l^j, L_l, f_l)$$

(2) Phase de correction par passage à une grille grossière

- Calcul du défaut : $\bar{d}_l^j := f_l - L_l u_l^j$

- Restriction du défaut : $\bar{d}_{l-1}^j := I_l^{l-1} \bar{d}_l^j$

- Calcul d'une approximation $\hat{v}_{l-1}^j$ de l'équation du défaut sur $\Omega_{l-1}$

$$L_{l-1} \hat{v}_{l-1}^j = \bar{d}_{l-1}^j$$

en appliquant $\gamma \geqslant 1$ itérations d'une méthode $l$-grilles (utilisant les grilles $\Omega_{l-1}$, $\Omega_{l-2}$, $\cdots$, $\Omega_0$ et les opérateurs de grille correspondants) et avec la fonction de grille nulle pour première approximation.

- Interpolation de la correction : $\tilde{v}_l^j := I_{l-1}^l \hat{v}_{l-1}^j$

- Calcul de l'approximation corrigée sur $\Omega_l$ : $\bar{u}_l^j + \tilde{v}_l^j$

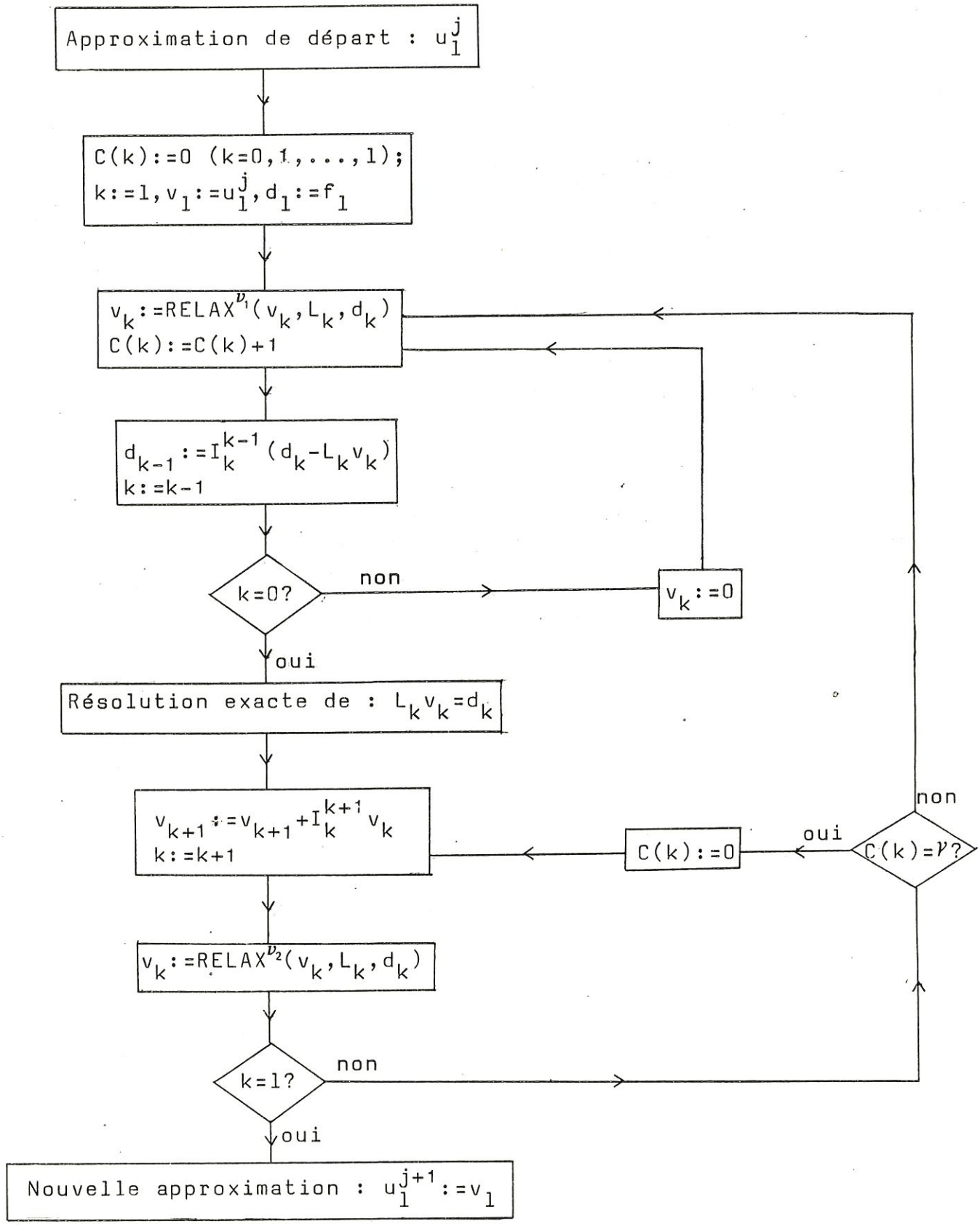
(3) Phase de lissage

- Calcul de $u_l^{j+1}$ en appliquant $\nu_2$ $(\geqslant 0)$ pas d'une méthode de relaxation donnée à $\bar{u}_l^j + \tilde{v}_l^j$ :

$$u_l^{j+1} := \mathrm{RELAX}^{\nu_2}(\bar{u}_l^j + \tilde{v}_l^j, L_l, f_l)$$

La même procédure est décrite dans l'organigramme suivant. Remarquons qu'un paramètre de contrôle $C(k)$ où $0 \leqslant C(k) \leqslant \gamma$ est introduit afin de décider quand il faut passer à une grille plus grossière où au contraire quand il faut retourner à une grille fine.

ORGANIGRAMME D'UNE ITERATION MULTIGRILLE POUR LA RESOLUTION DE $L_h u_h = f_h$ $(l \geq 1)$



## Matrice d'itération d'un cycle multigrille complet

### Proposition 3.1

La matrice d'itération $M_1$ de la méthode multigrille décrite précédemment est donnée par la récurrence

$$(a) \qquad M_1 = S_1^{\nu_2} (I_1 - I_0^1 L_0^{-1} I_1^0 L_1) S_1^{\nu_1} \qquad (3.1)$$

$$(b) \qquad M_{k+1} = S_{k+1}^{\nu_2} (I_{k+1} - I_k^{k+1} (I_k - M_k^{\nu}) L_k^{-1} I_{k+1}^k L_{k+1}) S_{k+1}^{\nu_1} \qquad (3.2)$$

pour $k = 1, 2, \ldots, l-1$

### Démonstration

(a) Nous avons vu au début de ce chapitre que la matrice d'itération de la méthode deux-grilles était donnée par :

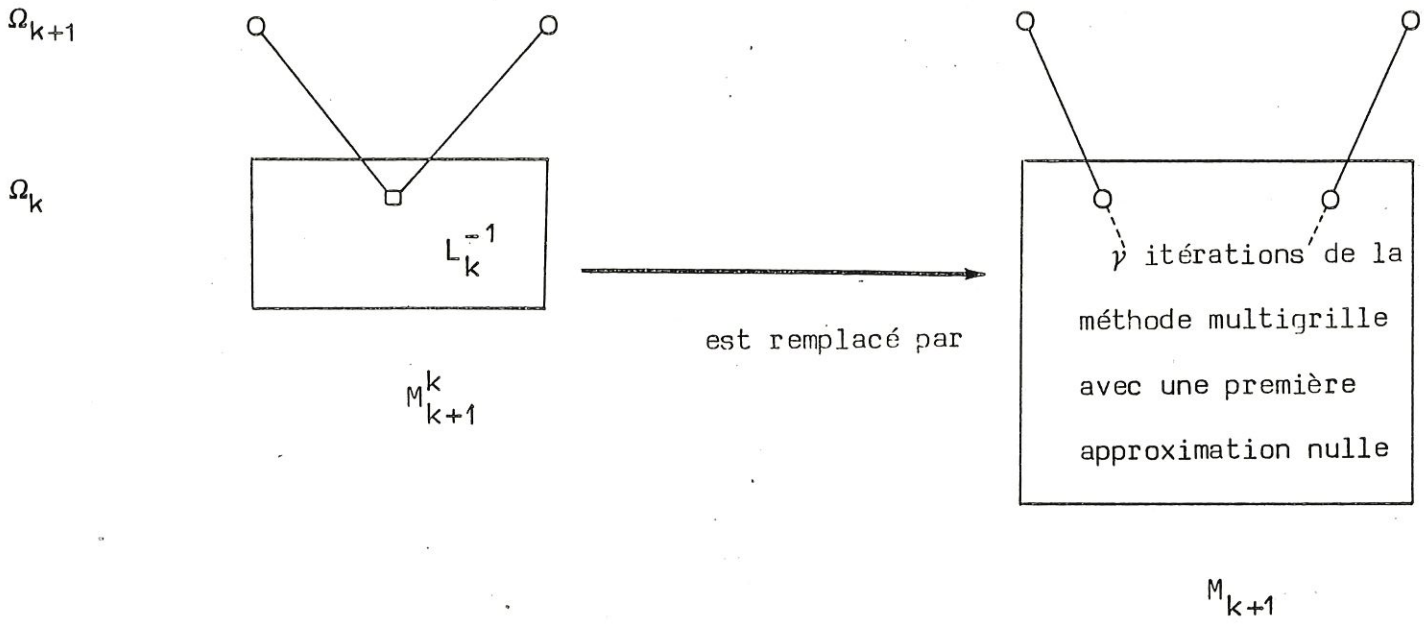
$$M_h^H = S_h^{\nu_2} (I_h - I_H^h L_H^{-1} I_h^H L_h) S_h^{\nu_1}$$

Réécrivons cette formule en utilisant les notations introduites précédemment : $h_1$ au lieu de $h$ et $h_0$ au lieu de $H$

Nous obtenons bien la forme (3.1) :

$$M_1 = M_1^0 = S_1^{\nu_2} (I_1 - I_0^1 L_0^{-1} I_1^0 L_1) S_1^{\nu_1}$$

(b) Nous avons vu lors de la description de la méthode multi grille qu'une telle méthode consistait à remplacer la résolution directe de l'équation du défaut sur la grille grossière par $\gamma$ itérations de la méthode multi grille elle-même c'est-à-dire :




Il s'agit donc, dans la formule connue :

$$M_{k+1}^k = S_{k+1}^{\nu_2} \left(I_{k+1} - I_k^{k+1} L_k^{-1} I_{k+1}^k L_{k+1}\right) S_{k+1}^{\nu_1} \qquad (3.3)$$

de remplacer l'opérateur $L_k^{-1}$ par un opérateur représentant $\gamma$ itérations de la méthode multigrille avec une première approximation nulle.

Or, si un système non singulier d'équations linéaires $L_k \hat{v}_k = \bar{d}_k$ est résolu approximativement par $\gamma$ pas d'une méthode itérative :

$\hat{v}_k^{j+1} = M\hat{v}_k^j + c$ avec $\hat{v}_k^0 = 0$, alors le $\gamma^{\text{ième}}$ itéré peut être représenté comme :

$$\hat{v}_k^{\gamma} = (I - M^{\gamma}) L_k^{-1} \bar{d}_k^j$$

En effet :

$$\hat{v}_k^1 = M\hat{v}_k^0 + c = c$$

$$\hat{v}_k^2 = Mc + c$$

$$\hat{v}_k^3 = M(Mc + c) + c = M^2c + Mc + c$$

$$\vdots$$

$$\hat{v}_k^\nu = M^{\nu-1}c + \ldots + Mc + c$$

$$= (M^{\nu-1} + \ldots + M + I)c$$

$$= (I - M^\nu)(I - M)^{-1}c$$

$$= (I - M^\nu)(I - M)^{-1}(I - M)L_k^{-1}\bar{d}_k$$

$$= (I - M^\nu)L_k^{-1}\bar{d}_k$$

En remplaçant dans la formule (3.3) $L_k^{-1}$ par la valeur ci-dessus, nous obtenons :

$$M_{k+1} = S_{k+1}^{\nu_2}(I_{k+1} - I_k^{k+1}(I - M_k^\nu)L_k^{-1}I_{k+1}^k L_{k+1})S_{k+1}^{\nu_1}$$

Ce qui démontre la proposition.

Nous allons à présent montrer qu'il est possible d'exprimer l'opérateur $M_k$ comme une perturbation de l'opérateur deux-grilles $M_k^{k-1}$ .

En effet, grâce à la proposition(3.3), nous avons le

Corollaire 3.1.

Pour $k = 1, \ldots, l-1$ , on a :

$$M_{k+1} = M^k_{k+1} + A^{k+1}_k M^{\nu}_k A^k_{k+1}$$

où

$$A^{k+1}_k := S^{\nu_2}_{k+1} I^{k+1}_k : \mathbb{G}(\Omega_k) \to \mathbb{G}(\Omega_{k+1})$$

$$A^k_{k+1} := L^{-1}_k I^k_{k+1} L_{k+1} S^{\nu_1}_{k+1} : \mathbb{G}(\Omega_{k+1}) \to \mathbb{G}(\Omega_k)$$

Preuve

De la proposition (3.3), il résulte que

$$M_{k+1} = S^{\nu_2}_{k+1} (I_{k+1} - I^{k+1}_k (I_k - M^{\nu}_k) L^{-1}_k I^k_{k+1} L_{k+1}) S^{\nu_1}_{k+1}$$

pour $k = 1, \ldots, l-1$

$$= S^{\nu_2}_{k+1} (I_{k+1} - I^{k+1}_k L^{-1}_k I^k_{k+1} L_{k+1}) S^{\nu_1}_k + S^{\nu_2}_{k+1} I^{k+1}_k M_k L^{-1}_k I^k_{k+1} L_{k+1} S^{\nu_1}_{k+1}$$

où l'on reconnait la forme de l'opérateur deux-grilles vue précédemment, soit

$$M^k_{k+1} = S^{\nu_2}_{k+1} (I_{k+1} - I^{k+1}_k L^{-1}_k I^k_{k+1} L_{k+1}) S^{\nu_1}_k$$

Posons :

$$A^{k+1}_k = S^{\nu_2}_{k+1} I^{k+1}_k$$

$$A^k_{k+1} = L^{-1}_k I^k_{k+1} L_{k+1} S^{\nu_1}_{k+1}$$

Il vient :

$$M_{k+1} = M^k_{k+1} + A^{k+1}_k M^{\nu}_k A^k_{k+1}$$

ce qui termine la preuve de ce corollaire.

Cette représentation de l'opérateur $M_k$ comme une perturbation de l'opérateur deux-grilles $M_{k+1}^k$ nous permet de donner une estimation de la norme de $M_1$ lorsque une estimation de la norme des opérateurs $M_{k+1}^k$ , $A_k^{k+1}$ , $A_{k+1}^k$ $(k \leqslant l-1)$ est connue à priori.

En effet on a la proposition :

Proposition 3.2

Si $\|M_{k+1}^k\| \leqslant \sigma^*$ , $\|A_k^{k+1}\| \, \|A_{k+1}^k\| \leqslant C$ pour tout $k \leqslant l-1$

Alors on a :

$$\|M_1\| \leqslant \eta_1$$

où $\eta_1$ est défini par :

$$\eta_1 := \sigma^*$$

$$\eta_{k+1} := \sigma^* + C\,\eta_k^{\nu} \qquad k = 1, \ldots, l-1$$

La preuve de cette proposition se fait par récurrence :

Nous avons vu (démonstration de la proposition ( 3.1)) que :

$$M_1 = M_1^0$$

d'où l'on tire :

$$\|M_1\| = \|M_1^0\| \leqslant \sigma^* = \eta_1$$

Reprenons l'expression de $M_k$ comme une perturbation de $M^k_{k+1}$

$$M_{k+1} = M^k_{k+1} + A^{k+1}_k M^{\nu}_k A^k_{k+1}$$

ce qui nous donne :

$$\|M_{k+1}\| \leqslant \|M^k_{k+1}\| + \|A^{k+1}_k\| \|A^k_{k+1}\| \|M_k\|^{\nu}$$

$$\leqslant \sigma^* + C \eta_k^{\nu}$$

$$= \eta_{k+1}$$

En particulier, il vient pour $k = l-1$ :

$$\|M_1\| \leqslant \eta_1$$

avec $\eta_k$ , $k = 1, \ldots, l-1$ définis comme ci-dessus.

Ce qui termine la démonstration de la proposition (3.2).

Cette proposition nous permet d'obtenir pour le cas $\nu = 2$, l'estimation suivante de la norme de l'opérateur $M_1$ :

Corollaire 3.2

Dans le cas où $\nu = 2$ et si $4C\sigma^* \leqslant 1$ , nous obtenons l'estimation :

$$\|M_1\| \leqslant \eta := \left(1 - \sqrt{1 - 4C\sigma^*}\right) / 2C \leqslant 2\sigma^* \qquad l \geqslant 1 \qquad (3.4)$$

où la borne $\eta$ est indépendante de $l$ .

En effet, supposons que la récurrence

$$\eta_{k+1} = \sigma^* + C\,\eta_k^2$$

$$\eta_1 = \sigma^*$$

converge, il vient :

$$\eta = \sigma^* + C\,\eta^2 \iff C\,\eta^2 - \eta + \sigma^* = 0$$

d'où les solutions $\eta_{1,2} = \dfrac{1 \pm \sqrt{1-4C\sigma^*}}{2C}$

Comme $4C\sigma^* < 1$ , les solutions sont réelles.

En outre on peut montrer aisément, en utilisant le principe de contraction, que la récurrence converge sur l'intervalle $[0, \frac{1}{2C}]$ ce qui conduit à considérer la racine

$$\eta = \frac{1 - \sqrt{1 - 4C\sigma^*}}{2C}$$

d'où la thèse.

Remarquons que pour $\sigma^*$ suffisamment petit, nous obtenons $\eta \approx \sigma^*$ pour la borne $\eta$ donné dans ( 3.4 )

Par exemple, si $C = 1$ , nous tirons de ( 3.4 ) :

$$\eta \leqslant 0.113 \text{ si } \sigma^* \leqslant 0.1$$

(la constante C est toujours $\geqslant 1$ et proche de 1. Par exemple, pour la méthode traitée dans la section 3.3, nous avons $C = 1$)

En conséquence, si la méthode deux-grilles converge suffisamment bien ( $\sigma^*$ petit), alors la méthode multigrille correspondante avec $\gamma = 2$ possède les mêmes propriétés de convergence.

Ainsi pour le problème considéré il suffit, pour ce qui touche la convergence, de ne regarder que l'étape deux-grilles.

Pour terminer, nous allons montrer que la méthode multigrille est asymptotiquement optimale dans la mesure où le nombre d'opérations requises pour résoudre le système $L_h u_h = f_h$ est proportionnel au nombre de points de grilles, la constante de proportionnalité étant petite.

Efficacité de la méthode et volume de calcul nécessaire

(Par volume de calcul, nous entendons une mesure du nombre des opérations arithmétiques)

Le travail effectué au cours d'un cycle multigrille peut s'écrire, compte tenu du corollaire ( 3.1 ) :

$$W_1 = W_1^0 + W_0 \qquad (3.5)$$

$$W_{k+1} = W_{k+1}^k + \gamma_k W_k \qquad (3.6)$$

où $W_0$ désigne le volume des opérations nécessaires pour calculer la solution exacte sur la grille grossière.

$W_{k+1}^k$ désigne le volume des opérations nécessaires pour résoudre un cycle deux-grilles ( $h_{k+1}$ , $h_k$ )

$\gamma_k$ est le nombre d'itérations du cycle deux-grilles ( $h_{k+1}$ , $h_k$ ).

De ( 3.5 ) et ( 3.6 ) on tire en supposant $\gamma_k = \gamma$ :

$$W_1 = \gamma^{l-1} W_1^0 + \gamma^{l-2} W_2^1 + \ldots + \gamma^0 W_1^{l-1} + \gamma^{l-1} W_0$$

$$= \sum_{k=1}^{l} \gamma^{l-k} W_k^{k-1} + \gamma^{l-1} W_0 \qquad (3.7)$$

Estimation du nombre d'opérations arithmétiques nécessaires pour effectuer un cycle deux-grilles ( $h_{k+1}$ , $h_k$ ).

Si nous désignons par $C$ le nombre des opérations réclamées en chaque point par les composantes du cycle :

- relaxation
- calcul des défauts
- passage de la grille fine à la grille grossière
- et inversément,

il vient dans le cas où toutes les composantes multigrilles sont construites de la même façon sur toutes les grilles (ce qui est le cas dans le problème que nous étudions (cfr § 3.4.1) )

$$w_k^{k+1} \doteq C \mathcal{N}_k$$

où $\mathcal{N}_k$ désigne le nombre de points de la grille $\mathcal{N}_k$ ( $\mathcal{N}_k = \# \Omega_k$ )

$\doteq$ est une notation signifiant l'égalité à des effets d'ordre inférieur près (comportement aux frontières)

Nous pouvons estimer la constante $C$ de manière plus précise en prenant pour conventions :

- $\nu = \nu_1 + \nu_2$ représente le nombre total de relaxations au cours d'un cycle
- $w_0$ , $w_1$ , $w_2$ sont des mesures du nombre d'opérations nécessaires par point de la grille $\Omega_k$ pour les différents éléments qui composent un cycle deux-grilles, à savoir respectivement :

- $w_0$ : une relaxation sur $\Omega_k$
- $w_1$ : calcul du défaut et transfert de $\Omega_k$ vers $\Omega_{k-1}$
- $w_2$ : interpolation de la correction de $\Omega_{k-1}$ vers $\Omega_k$ et son ajoût à l'approximation précédente.

Il vient

$$W_k^{k-1} \doteq (\nu w_0 + w_1 + w_2)\ \mathcal{N}_k$$

Dans notre cas $w_0$ , $w_1$ et $w_2$ ne dépendent pas de k et peuvent être estimés ([11]) à :

| | $w_0$ | $w_1$ | $w_2$ | $W_k^{k-1}$ |
|---|---|---|---|---|
| +/- | 5 | 25/4 | 7/4 | $(5\nu + 8)\mathcal{N}_k$ |
| * | 1 | 5/4 | 3/4 | $(\nu + 2)\mathcal{N}_k$ |

En comptabilisant de la même manière les additions et les multiplications, il vient :

$$C = (6\nu + 10)$$

d'où $$W_k^{k-1} \doteq (6\nu + 10)\mathcal{N}_k$$

Le travail nécessaire pour effectuer un cycle complet est alors donné par :

$$W_1 = C\,(\nu^{l-1}\ \mathcal{N}_1 + \nu^{l-2}\ \mathcal{N}_{l-1} + \ldots + \nu^0\ \mathcal{N}_l) + \nu^{l-1}\ W_0$$

$$= C\ \mathcal{N}_1\ (\nu^{l-1} + \frac{\nu^{l-2}}{2^2} + \frac{\nu^{l-3}}{2^4} + \ldots + \frac{\nu^0}{2^{2(l-1)}}) + \nu^{l-1} W_0$$

Si $\nu = 1$

$$W_1 = C\,\mathcal{N}_1 \left( 1 + \left(\frac{1}{2^2}\right)^1 + \left(\frac{1}{2^2}\right)^2 + \left(\frac{1}{2^2}\right)^3 + \ldots \right) + W_0$$

$$= C\,\mathcal{N}_1 \left( \frac{(1/2^2)^1 - 1}{(1/2^2) - 1} \right) + W_0$$

$$\doteq C\,\mathcal{N}_1 \frac{4}{3} + W_0$$

d'où $$W_1 = \frac{4}{3} C\,\mathcal{N}_1 + W_0$$

En particulier, dans le programme que nous avons travaillé :

$W_0 \approx 0$ (cfr § 3.4.1.)

D'où $$W_1 = \frac{4}{3} C\,\mathcal{N}_1$$

Nous obtenons des résultats analogues pour $\nu = 2$ et 3 ce qui nous permet de dresser le tableau récapitulatif ci-dessous :

$$W_1 \doteq \begin{cases} \frac{4}{3} C\,\mathcal{N}_1 & (\text{pour } \nu = 1) \\ 2\,C\,\mathcal{N}_1 & (\text{pour } \nu = 2) \\ 4\,C\,\mathcal{N}_1 & (\text{pour } \nu = 3) \end{cases}$$

## 3.2. METHODE MULTIGRILLE UNIDIMENSIONNELLE

### 3.2.1 Problème modèle

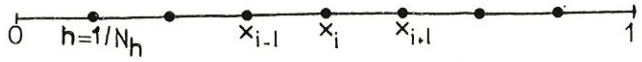
Nous allons exposer la méthode en prenant comme modèle l'équation de Poisson avec les conditions aux limites de Dirichlet :

$$\begin{cases} -\dfrac{d^2u}{dx^2} = f & \text{sur } \Omega = (0,1) \\ u = g & \text{sur } \Gamma = \partial\Omega = \{0,1\} \end{cases} \qquad (3.8)$$

### 3.2.2 Formulation discrète du problème

Soit $\Omega_h = \left\{ \dfrac{1}{N_h}, \dfrac{2}{N_h}, \ldots, \dfrac{N_h-1}{N_h} \right\}$

l'équivalent discret de pas h du domaine $\Omega$ où $N_h = 2^m$ est fixé




et soit $\Omega_H = \Omega_{2h} = \left\{ \dfrac{2}{N_h}, \dfrac{4}{N_h}, \ldots, \dfrac{2(N_h-1)}{N_h} \right\} \subset \Omega_h$

En approchant la dérivée seconde $\left(\frac{d^2u}{dx^2}\right)_i$ au point i par l'expression

$$\frac{u_{i+1} - 2u_i + u_{i-1}}{h^2} \qquad \text{où} \quad u(x_i) := u_i$$

encore notée

$$\frac{1}{h^2}\,[1 \quad -2 \quad 1]\,u_h$$

on obtient la forme discrète de l'équation de Poisson

$$\frac{1}{h^2}\,[1 \quad -2 \quad 1]\,u_h^j = f_h \qquad (3.9)$$

avec les conditions aux limites

$$\begin{cases} u_h(0) = g(0) \\ \\ u_h(1) = g(1) \end{cases} \qquad (3.10)$$

en incluant les relations (3.10) dans le système (3.9), on obtient un système matriciel qui, dans le cas particulier où $N_h = 8$ peut s'écrire :

$$\frac{1}{h^2}\begin{bmatrix} 2 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 2 \end{bmatrix} \begin{bmatrix} u(h) \\ u(2h) \\ u(3h) \\ u(4h) \\ u(5h) \\ u(6h) \\ u(7h) \end{bmatrix} = \begin{bmatrix} f(h) + \frac{1}{h^2}\,g(0) \\ f(2h) \\ f(3h) \\ f(4h) \\ f(5h) \\ f(6h) \\ f(7h) + \frac{1}{h^2}\,g(1) \end{bmatrix}$$

Ce système sera noté sous forme condensée :

$$L_h \, u_h = f_h \qquad (3.11)$$

On observe tout d'abord que $L_h$ est une matrice symétrique dont les vecteurs propres s'écrivent

$$\varphi^n(x) = \sin(n\pi x) \qquad x \in \Omega_h \quad , \quad n = 1, \ldots N_h - 1$$

où

$$\varphi^n(x) = \begin{bmatrix} \varphi^n(1/N_h) \\ \varphi^n(2/N_h) \\ \varphi^n(3/N_h) \\ \varphi^n(4/N_h) \\ \varphi^n(5/N_h) \\ \varphi^n(6/N_h) \\ \varphi^n(7/N_h) \end{bmatrix}$$

Les valeurs propres sont positives et égales à :

$$\frac{2}{h^2} \left(1 - \cos n\pi h\right) \qquad n = 1, \ldots N_h - 1$$

En effet :

$$L_h \varphi^n = \frac{2}{\omega h^2} \left( 1 - (1 - \omega + \omega \cos n\pi h) \right) \varphi^n$$

$$= \frac{2}{\omega h^2} \omega (1 - \cos n\pi h) \varphi^n$$

$$= \frac{2}{h^2} (1 - \cos n\pi h) \varphi^n \qquad n = 1, \ldots, N_h - 1$$

On en déduit que la solution $u_h$ de $L_h u_h = f_h$ peut s'écrire sous la forme :

$$u_h = \sum_{i=1}^{N_h - 1} \alpha_i \varphi^i(x)$$

Nous aurons besoin de ce résultat dans la suite.

Nous allons maintenant détailler la méthode multigrille décrite dans l'introduction ce qui revient tout d'abord à analyser chaque composante de la méthode deux-grilles.

## 3.2.3. Méthode deux-grilles

### a. Définition de l'opérateur de lissage

Appliquons la procédure de Jacobi au système

$$L_h u_h := \frac{1}{h^2} [-1 \quad 2 \quad -1] \; u_h = f_h$$

nous obtenons en remplaçant $L_h$ par sa composante diagonale

$$\frac{2}{h^2} u_h^{j+1} + \frac{1}{h^2}[-1 \quad 0 \quad -1] \; u_h^j = f_h$$

ce qui donne

$$u_h^{j+1} = \frac{1}{2} [1 \quad 0 \quad 1] \; u_h^j + \frac{h^2}{2} \frac{1}{h^2} [-1 \quad 2 \quad -1] \; u_h^j$$

en posant $v_h^j = u_h - u_h^j$, il vient

$$v_h^{j+1} = \frac{1}{2} [1 \quad 0 \quad 1] \; v_h^j$$

Et en introduisant un paramètre de relaxation $\omega$ $(0 \leqslant \omega \leqslant 1)$, nous obtenons

$$v_h^{j+1} = \frac{\omega}{2} [1 \quad 2(\frac{1}{\omega} - 1) \quad 1] \; v_h^j$$

où $\frac{\omega}{2} [1 \quad 2(\frac{1}{\omega} - 1) \quad 1] := S_h(\omega)$ est, par définition, l'opérateur de lissage.

Cet opérateur peut encore s'exprimer sous la forme :

$$S_h(\omega) = \frac{\omega}{2} [1 \quad \frac{2}{\omega} - 2 \quad 1]$$

$$= [0 \quad 1 \quad 0] + \frac{\omega}{2} [1 \quad -2 \quad 1]$$

$$= I_h - \frac{\omega}{2} h^2 L_h$$

Il admet les mêmes vecteurs propres que $L_h$ :

$$\varphi^n(x) = \sin(n\pi x) \qquad n = , \ldots, N_h - 1$$

$$x \in \Omega_h$$

Ses valeurs propres sont égales à

$$K_n(\omega) = 1 - \omega + \omega \cos n\pi h \qquad n = 1, \ldots, N_h - 1$$

En effet

$$S_h(\omega)\ \varphi^n = (I_h - \frac{\omega}{2} h^2 L_h)\ \varphi^n$$

$$= (\ 1 - \frac{\omega}{2} h^2 \frac{2}{h^2} (\ 1 - \cos n\pi h\ ))\ \varphi^n$$

$$= (\ 1 - \omega + \omega \cos n\pi h\ )\ \varphi^n$$

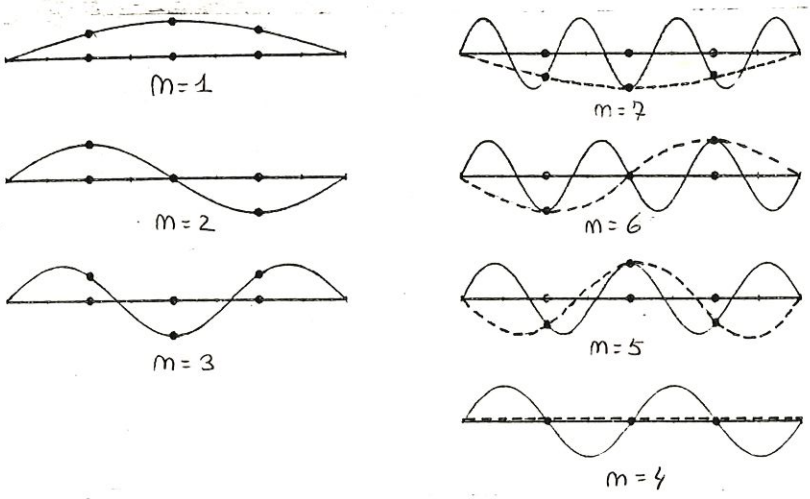
$$n = 1, \ldots, N_h - 1$$

Remarquons dès à présent comment sont affectés les vecteurs propres $\varphi^n(x) = \sin(n\pi x)$ $\quad x \in \Omega_h, n=1, \ldots N_h-1$ par le passage de la grille fine $\Omega_h$ à la grille grossière $\Omega_{2h}$ .

Examinons le cas où $N_h = 8$

$$\varphi^n(x) = \sin(n\pi x) \qquad x \in \left\{\frac{1}{8}, \frac{2}{8}, \frac{3}{8}, \ldots, \frac{7}{8}\right\}$$

$$n = 1, 2, \ldots, 7$$



Les vecteurs propres $\varphi^n$ pour $n = 1,2,3$ sont aussi représentables sur la grille grossière $\Omega_{2h}$.
Nous les appellerons vecteurs propres "basse fréquence".

Les vecteurs propres $\varphi^n$ pour $n = 4,5,6,7$ ne sont pas "représentables" sur la grille grossière $\Omega_{2h}$
En particulier la représentation de $\varphi^4$ sur la grille grossière $\Omega_{2h}$ est la fonction nulle.
Ces vecteurs propres sont dits "haute fréquence".

Nous avons constaté que toute solution $u_h$ de $L_h u_h = f_h$ peut s'exprimer comme une combinaison linéaire des vecteurs propres de l'opérateur $L_h$

$$u_h = \sum_{n=1}^{N_h-1} \alpha_n \varphi^n$$

ou en remplaçant les vecteurs propres $\varphi^n$ par leurs valeurs :

$$u_h = \sum_{n=1}^{N_h-1} \alpha_n \sin(n\pi x) \qquad x \in \Omega_h$$

Puisque les opérateurs $L_h$ et $S_h(\omega)$ possèdent les mêmes vecteurs propres, nous pouvons exprimer $S_h u_h$ comme :

$$S_h u_h = \sum_{n=1}^{N_h-1} \alpha_n S_h \sin(n\pi x)$$

$$= \sum_{n=1}^{N_h-1} \alpha_n (1 - \omega + \omega \cos(n\pi h)) \sin(n\pi x)$$

Nous remarquons que les composantes haute fréquence de la solution $u_h$ sont réduites.

Si nous prenons $\omega = \frac{2}{3}$ par exemple, ces composantes seront au moins réduites d'un facteur

$$\mu(h, S_h(\tfrac{2}{3})) = \max_{\frac{N_h}{2} \leqslant n \leqslant N_h-1} \left| \frac{1}{3} + \frac{2}{3} \cos(n\pi h) \right|$$

$$= \frac{1}{3}$$

à chaque itération de la méthode de Jacobi.

Cet effet de lissage est mesuré grâce au paramètre

$$\mu(h, S_h(\omega)) = \max_{\frac{N_h}{2} \leqslant n \leqslant N_h - 1} |1 - \omega + \omega \cos(n\pi h)|$$

appelé facteur de lissage.

A titre d'exemple, estimons ce facteur pour différentes valeurs du paramètre de relaxation $\omega$ :

$\underline{\omega = 1}$ : $K_n = \cos(n\pi h)$

$$\mu(h, S_h(\omega)) = \max_{\frac{N_h}{2} \leqslant n \leqslant N_h - 1} |\cos(n\pi h)| = |\cos((N_h - 1)\pi h)|$$

$$= \cos(\pi h)$$

$$= 1 - O(h^2)$$

La méthode de Jacobi utilisée avec un facteur de relaxation $\omega = 1$ possède de mauvaises propriétés de lissage.

$\underline{\omega = 1/2}$ : $K_n = \frac{1}{2} + \frac{1}{2} \cos(n\pi h)$

$$\mu(h, S_h(\omega)) = \max_{\frac{N_h}{2} \leqslant n \leqslant N_h - 1} \left| \frac{1}{2} + \frac{1}{2} \cos(n\pi h) \right| = \frac{1}{2}$$

La méthode de Jacobi utilisée avec un facteur de relaxation $\omega = \frac{1}{2}$ possède de bonnes propriétés de lissage : les composantes haute fréquence de la solution seront au moins réduites d'un facteur $\frac{1}{2}$ par itération de Jacobi.

Lemme 3.1

Si $h$ est suffisamment petit, nous avons :

$$\mu(h, S_h(\omega)) = \begin{cases} 1 - \omega & \text{si } \omega \leq 2/3 \\ \omega - 1 - \omega \cos(\pi h) & \text{si } \omega > 2/3 \end{cases}$$

En outre, la valeur $\omega = 2/3$ est optimale et conduit donc au plus petit facteur de lissage :

$$\mu(h, S_h(2/3)) = 1/3.$$

Preuve :

Rappelons que le facteur de lissage $\mu$ est donné par l'expression :

$$\mu = \max_{\frac{N_h}{2} \leq n \leq N_h - 1} \left| 1 - \omega + \omega \cos(n\pi h) \right|$$

Afin de tracer le graphe de cette fonction dépendant de $\omega$, on analyse séparémment :

$$\mu^1 = |1 - \omega + \omega \cos(N_h - 1)\pi h|$$

et $$\mu^2 = |1 - \omega + \omega \cos \frac{N_h}{2} \pi h|$$

en observant que $\mu = \max\{\mu^1, \mu^2\}$

Etude de $\mu^1$ $= |1 - \omega + \omega \cos(N_h - 1)\pi h|$

Il vient :

$$\cos(N_h - 1)\pi h = -\cos\pi h = -1 + O(h^2)$$

d'où

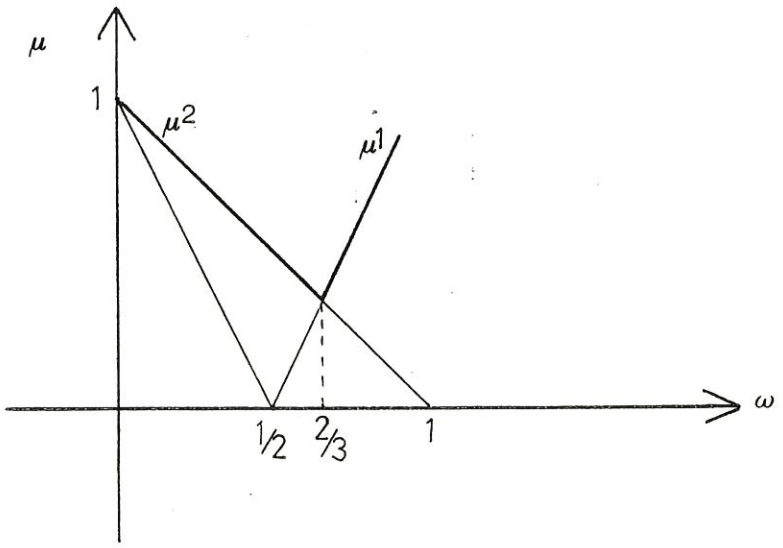
$$\mu^1 = |1 - 2\omega + O(h^2)|$$

et aux termes du second ordre près :

$$\mu^1 = \begin{cases} 2\omega - 1 & \text{si} \quad \omega \geqslant 1/2 \\ 1 - 2\omega & \text{si} \quad \omega < 1/2 \end{cases}$$

Etude de $\mu^2$ $= |1 - \omega| = 1 - \omega$

La représentation graphique de $\mu^1$ et de $\mu^2$ met en évidence le point critique $\omega = 2/3$



b. Définition de l'opérateur de correction par grille grossière (CGC)

Nous avons vu que cette méthode remplace la résolution du système :

$$L_h \, v_h^j = d_h^j \quad , \; j = 0,1,2,\ldots$$

sur la grille fine $\Omega_h$

par la résolution de :

$$L_H \, \hat{v}_H^j = d_H^j \quad , \; j = 0,1,2,\ldots$$

sur la grille grossière $\Omega_H$

En plus d'une méthode de résolution pour le système $L_H \, \hat{v}_H^j = d_H^j$, nous aurons donc besoin de deux opérateurs de transfert :

un opérateur de restriction $I_h^{2h}$ et un opérateur de prolongement $I_{2h}^h$

1- L'opérateur de restriction : $I_h^{2h}$

envoie le résidu donné dans $\Omega_h$ soit $d_h^j$ sur sa restriction dans $\Omega_{2h}$ : $d_{2h}^j$

Cet opérateur doit être simple à calculer tout en perdant le moins d'information possible lors du transfert de grille à grille.

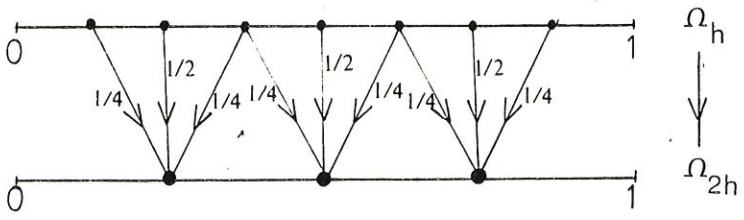
Nous choisissons

$$I_h^{2h} = \frac{1}{4} \begin{bmatrix} 1 & 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 2 & 1 \end{bmatrix}$$

qui correspond à la molécule :

$$\frac{1}{4} \begin{bmatrix} 1 & 2 & 1 \end{bmatrix}$$

ou graphiquement :



2- A l'inverse, l'opérateur de prolongement : $I_{2h}^{h}$ — envoie l'erreur $v_{2h}^{j}$ calculée dans $\Omega_{2h}$ (qui est solution du système $L_{2h}\, v_{2h}^{j} = d_{2h}^{j}$ ) sur sa prolongée dans $\Omega_h$ : $v_h^j$

De nouveau nous désirons un opérateur simple à calculer et perdant un minimum d'informations.

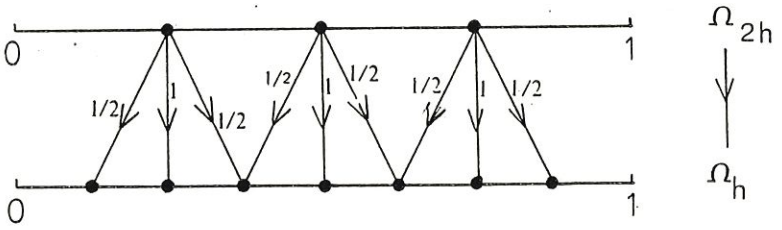
Nous choisissons

$$I_{2h}^{h} = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 2 \\ 0 & 0 & 1 \end{bmatrix}$$

qui correspond à la molécule

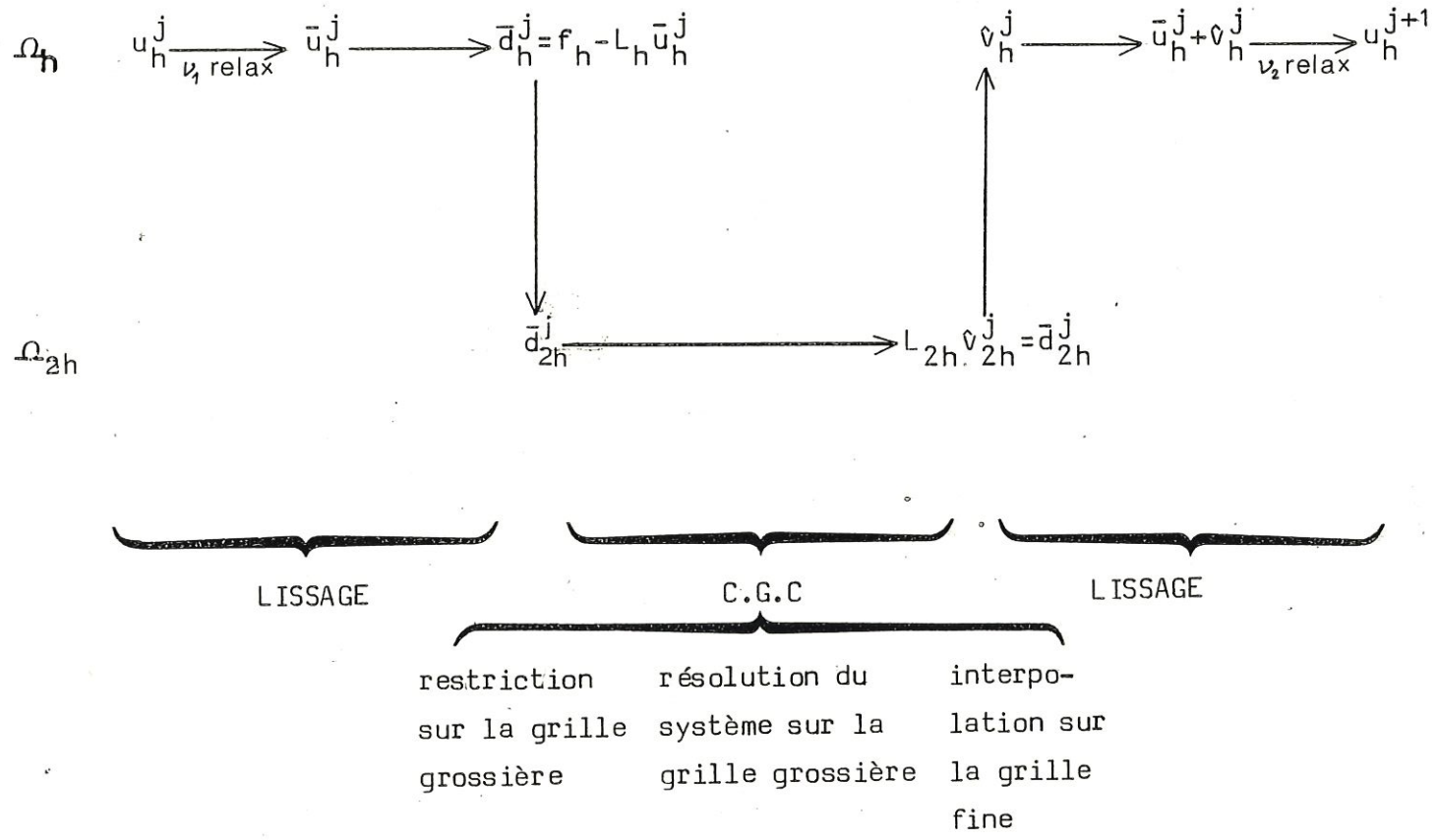
$$\frac{1}{2} \quad ]1 \quad 2 \quad 1[$$

ou graphiquement



c. Définition de l'opérateur d'itération deux-grilles

Combinons les opérations de lissage, de transfert et de résolution vues dans le chapitre précédent suivant le schéma "deux-grilles" exposé au chapitre 2.1




Une itération "deux-grilles" est alors caractérisée par l'opérateur matriciel :

$$M_h^{2h} = S_h^{\nu_2} \left(I_h - I_{2h}^h L_{2h}^{-1} I_h^{2h} L_h\right) S_h^{\nu_1}$$

En posant $K_h^{2h} = I_h - I_{2h}^h L_{2h}^{-1} I_h^{2h} L_h$ , il vient :

$$M_h^{2h} = S_h^{\nu_2} K_h^{2h} S_h^{\nu_1}$$

Pour caractériser les propriétés de convergence de l'opérateur $M_h^{2h}$, nous devons calculer son rayon spectral. Comme il est trop compliqué de calculer directement celui-ci, nous allons construire $\hat{M}_h^{2h}$, un équivalent orthogonal de la matrice $M_h^{2h}$ simple à manipuler.
Afin de simplifier les calculs nous prendrons $N_h = 8$.

Observons tout d'abord que :

$$\Phi^n(x) := \varphi^n(x) = -\varphi^{N_h - n}(x) \qquad x \in \Omega_{2h}$$

$$\Phi^{N_h/2}(x) := \varphi^{N_h/2}(x) = 0 \qquad x \in \Omega_{2h}$$

En effet :

$$\varphi^{N_h - n}(x) = -\sin((N_h - n)\pi x)$$
$$= -\sin(N_h \pi x)\cos(n\pi x) + \sin(n\pi x)\cos(N_h \pi x)$$
$$= \sin(n\pi x)$$
$$= \varphi^n(x)$$
où $x = lh$ avec $l = 2,4,6.$

$$\varphi^{N_h/2}(x) = \sin\left(\frac{N_h}{2}\pi x\right)$$
$$= \sin\left(l\,\frac{\pi}{2}\right)$$
$$= 0 \qquad \text{où } x = lh \text{ avec } l = 2,4,6.$$

Posons $E_{h,n} := \text{span}\,[\varphi^n, \varphi^{N_h - n}]$ c'est-à-dire l'espace engendré par les vecteurs $\varphi^n$, $\varphi^{N_h - n}$ et $E_{h,\frac{N_h}{2}} = \text{span}\,[\varphi^{\frac{N_h}{2}}]$

Nous allons d'abord montrer que ces sous-espaces de $\mathbb{G}(\Omega_h)$ sont laissés invariants par l'opérateur $K_h^{2h}$, c'est-à-dire

$$K_h^{2h} : E_{h,n} \longrightarrow E_{h,n} \qquad \left(n \leqslant \frac{N_h}{2}\right)$$

En effet,

$L_h : E_{h,n} \quad E_{h,n}$ : car $\varphi^n$ et $\varphi^{N_h - n}$ sont des vecteurs propres de $L_h$

$I_h^{2h} : E_{h,n} \longrightarrow \text{span}\,[\Phi^n]$

En effet, si nous appliquons l'opérateur $I_h^{2h}$ au vecteur propre $\varphi^n$ où $n \leq \frac{N_h}{2}$ il vient,

$$\frac{1}{2}\begin{bmatrix} 1/2 & 1 & 1/2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 1 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1 & 1/2 \end{bmatrix} \begin{bmatrix} \sin(n\pi h) \\ \sin(2n\pi h) \\ \sin(3n\pi h) \\ \sin(4n\pi h) \\ \sin(5n\pi h) \\ \sin(6n\pi h) \\ \sin(7n\pi h) \end{bmatrix}$$

$$= \frac{1}{2}\begin{bmatrix} \frac{1}{2}(\sin(n\pi h) + \sin(3n\pi h)) + \sin(2n\pi h) \\ \frac{1}{2}(\sin(3n\pi h) + \sin(5n\pi h)) + \sin(4n\pi h) \\ \frac{1}{2}(\sin(5n\pi h) + \sin(7n\pi h)) + \sin(6n\pi h) \end{bmatrix}$$

$$= \frac{1}{2}\begin{bmatrix} \sin(2n\pi h)(\cos(n\pi h) + 1) \\ \sin(4n\pi h)(\cos(n\pi h) + 1) \\ \sin(6n\pi h)(\cos(n\pi h) + 1) \end{bmatrix}$$

$$= \frac{1}{2}(\cos(n\pi h) + 1)\begin{bmatrix} \sin(2n\pi h) \\ \sin(4n\pi h) \\ \sin(6n\pi h) \end{bmatrix}$$

Appliquons à présent ce même opérateur $I_h^{2h}$ au vecteur propre $\varphi^{N_h - n}$ où $n < \frac{N_h}{2}$ il vient

$$\frac{1}{2}\begin{bmatrix} \frac{1}{2} & 1 & \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 1 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & 1 & \frac{1}{2} \end{bmatrix} \begin{bmatrix} \sin((N_h-n)\pi h) \\ \sin((N_h-n)2\pi h) \\ \sin((N_h-n)3\pi h) \\ \sin((N_h-n)4\pi h) \\ \sin((N_h-n)5\pi h) \\ \sin((N_h-n)6\pi h) \\ \sin((N_h-n)7\pi h) \end{bmatrix}$$

$$= \frac{1}{2}\begin{bmatrix} \frac{1}{2} & 1 & \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 1 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & 1 & \frac{1}{2} \end{bmatrix} \begin{bmatrix} \sin(n\pi h) \\ -\sin(2\,n\pi h) \\ \sin(3\,n\pi h) \\ -\sin(4\,n\pi h) \\ \sin(5\,n\pi h) \\ -\sin(6\,n\pi h) \\ \sin(7\,n\pi h) \end{bmatrix}$$

$$= \frac{1}{2}\begin{bmatrix} \frac{1}{2}(\sin(n\pi h) + \sin(3\,n\pi h)) - \sin(2\,n\pi h) \\ \frac{1}{2}(\sin(3\,n\pi h) + \sin(5\,n\pi h)) - \sin(4\,n\pi h) \\ \frac{1}{2}(\sin(5\,n\pi h) + \sin(7\,n\pi h)) - \sin(6\,n\pi h) \end{bmatrix}$$

$$= \frac{1}{2}\begin{bmatrix} \sin(2\,n\pi h)(\cos(n\pi h) - 1) \\ \sin(4\,bn\pi h)(\cos(n\pi h) - 1) \\ \sin(6\,n\pi h)(\cos(n\pi h) - 1) \end{bmatrix}$$

$$= \frac{1}{2}(\cos n\pi h - 1)\begin{bmatrix} \sin(2\,n\pi h) \\ \sin(4\,n\pi h) \\ \sin(6\,n\pi h) \end{bmatrix}$$

$L_{2h}$ : span $[\Phi^n] \longrightarrow$ span $[\Phi^n]$ : car $\Phi^n$ sont des vecteurs propres de $L_{2h}$

$I_{2h}^{h}$ : span $[\Phi^n] \longrightarrow E_{h,n}$

En effet :

$$\begin{bmatrix} \frac{1}{2} & 0 & 0 \\ 1 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 1 & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 1 \\ 0 & 0 & \frac{1}{2} \end{bmatrix} \begin{bmatrix} \sin 2n\pi h \\ \sin 4n\pi h \\ \sin 6n\pi h \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{2} \sin 2n\pi h \\ \sin 2n\pi h \\ \frac{1}{2} \sin 2n\pi h + \frac{1}{2} \sin 4n\pi h \\ \sin 4n\pi h \\ \frac{1}{2} \sin 4n\pi h + \frac{1}{2} \sin 6n\pi h \\ \sin 6n\pi h \\ \frac{1}{2} \sin 6n\pi h \end{bmatrix}$$

$$= \alpha \begin{bmatrix} \sin n\pi h \\ \sin 2n\pi h \\ \sin 3n\pi h \\ \sin 4n\pi h \\ \sin 5n\pi h \\ \sin 6n\pi h \\ \sin 7n\pi h \end{bmatrix} + \beta \begin{bmatrix} \sin n\pi h \\ -\sin 2n\pi h \\ \sin 3n\pi h \\ -\sin 4n\pi h \\ \sin 5n\pi h \\ -\sin 6n\pi h \\ \sin 7n\pi h \end{bmatrix}$$

Pour les composantes paires ( l = 2,4,6), il vient :

$$\alpha \sin l n\pi h - \beta \sin l n\pi h = \sin l n\pi h$$

d'où $( \alpha - \beta ) = 1$

Pour les composantes impaires ( l = 1,3,5,7), il vient :

$$\alpha \sin l n\pi h + \beta \sin l n\pi h = \frac{1}{2} \sin ( l-1 ) n\pi h + \frac{1}{2} \sin ( l+1 ) n\pi h$$

d'où $( \alpha + \beta ) \sin l n\pi h = \cos n\pi h \sin l n\pi h$

et $\alpha + \beta = \cos n\pi h$

On en déduit les valeurs de $\alpha$ et $\beta$ :

$$\alpha = \frac{1}{2} (\cos n\pi h + 1 )$$

$$\beta = \frac{1}{2} (\cos n\pi h - 1 )$$

Nous pouvons maintenant construire $\hat{K}_h^{2h}$ l'équivalent orthogonal de $K_h^{2h}$ , formé des images par $K_h^{2h}$ des vecteurs de base de $\mathbb{G}(\Omega_h)$ pris dans l'ordre : $\varphi^1$, $\varphi^7$ ; $\varphi^2$ , $\varphi^6$ ; $\varphi^3$, $\varphi^5$ et $\varphi^4$.

Rappelons que $K_h^{2h} = I_h - I_{2h}^h L_{2h}^{-1} I_h^{2h} L_h$

et appliquons cet opérateur aux vecteurs $\varphi^n$ où $n \leqslant \frac{N_h}{2}$

$$(I_h - I_{2h}^h L_{2h}^{-1} I_h^{2h} L_h) \varphi^n$$

$$= \varphi^n - 2N_h^2 ( 1 - \cos n\pi h ) I_{2h}^h L_{2h}^{-1} I_h^{2h} L_h \varphi^n$$

$$= \varphi^n - 2N_h^2 ( 1 - \cos n\pi h ) \frac{1}{2} (\cos n\pi h + 1 ) I_{2h}^h L_{2h}^{-1} \Phi^n$$

$$= \varphi^n - N_h^2 ( 1 - \cos n\pi h ) ( 1 + \cos n\pi h ) \frac{2}{N_h^2 ( 1 - \cos 2n\pi h)} I_{2h}^h \Phi^n$$

$$= \varphi^n - \frac{2 \sin^2 n\pi h}{( 1 - \cos 2n\pi h )} ( \frac{1}{2} (\cos n\pi h + 1 ) \varphi^n + \frac{1}{2} (\cos n\pi h - 1 ) \varphi^{N_h - n} )$$

$$= \varphi^n - \cos^2 ( \frac{n\pi h}{2} ) \varphi^n + \sin^2 ( \frac{n\pi h}{2} ) \varphi^{N_h - n}$$

Posons $\sin^2 \frac{n\pi h}{2} = \xi$ d'où $\cos^2 \frac{n\pi h}{2} = 1 - \xi$

Il vient,

$$K_h^{2h} \varphi^n = \varphi^n - ((1-\xi)\varphi^n - \xi \varphi^{N_h - n})$$

En appliquant la même procédure à $\varphi^{N_h - n}$ où $n < \frac{N_h}{2}$ et à $\varphi^{N_h/2}$, nous obtenons

$$K_h^{2h} \varphi^{N_h - n} = \varphi^{N_h - n} - (-(1-\xi)\varphi^n + \xi \varphi^{N_h - n})$$

$$K_h^{2h} \varphi^{N_h/2} = \varphi^{N_h/2}$$

D'où l'on tire que l'équivalent orthogonal $\hat{K}_{2h}^h$ de $K_{2h}^h$ est diagonal par blocs $(\hat{K}_{2h,n}^h)$ où

$$\hat{K}_{h,n}^{2h} = \begin{cases} \begin{bmatrix} \xi & \xi - 1 \\ -\xi & 1 - \xi \end{bmatrix} & \text{si } n < \frac{N_h}{2} \\ \text{la matrice d'identité de dimension 1} & \text{si } n = \frac{N_h}{2} \end{cases}$$

D'autre part, comme les vecteurs $\varphi^n$ $(n=1, \ldots N_h - 1)$ sont aussi des vecteurs propres de $S_h$, on en déduit que les espaces $E_{h,n}$ $(n \leqslant \frac{N_h}{2})$ sont aussi laissés invariants par l'opérateur $S_h$ :

$$S_h : E_{h,n} \longrightarrow E_{h,n} \qquad n \leqslant \frac{N_h}{2}$$

L'équivalent diagonal $\hat{S}_h$ de $S_h$ se réduit à une diagonale constituée des valeurs propres $K_n = (1 - \omega + \omega \cos n\pi h)$, $n=1, \ldots N_h - 1$ de $S_h$ prises dans l'ordre $K_1, K_{N_h - 1}; K_2, K_{N_h - 2}; \ldots; K_{\frac{N_h}{2} - 1}, K_{\frac{N_h}{2} + 1}; K_{\frac{N_h}{2}}$

Les espaces $E_{h,n}$ $(n \leqslant N_h/2)$ étant laissés invariants par $K_h^{2h}$ et $S_h$, ils le sont aussi par $M_h^{2h}$. Et nous obtenons la représentation $\hat{M}_{h,n}^{2h}$ de $M_{h,n}^{2h}$ :

$$\hat{M}_{h,n}^{2h} = \hat{M}_{h,n}^{2h}(\nu_1, \nu_2; \omega) = \hat{S}_{h,n}(\omega)\, \hat{K}_{h,n}^{2h}\, \hat{S}_{h,n}(\omega)$$

Nous avons donc réduit le calcul du rayon spectral $\rho(M_h^{2h})$ au calcul des rayons spectraux de matrices (au plus) de dimension 2.

$$\rho(M_h^{2h}) = \max \left\{ \rho(\hat{M}_{h,n}^{2h}) : n \leqslant N_h/2 \right\}$$

Cette quantité dépend en particulier des paramètres $\omega$ , $\nu_1$ et $\nu_2$ ou plus exactement de $\omega$ et $\nu := \nu_1 + \nu_2$ .

En effet, comme $\rho(AB) = \rho(BA)$ où A et B sont des opérateurs linéaires, $\rho(M_h^{2h})$ ne dépend pas de $\nu_1$ et $\nu_2$ individuellement mais de leur somme ( $\nu$ ).

Pour la méthode développée dans ce chapitre, c'est-à-dire quand la technique de lissage utilisée est celle de Jacobi relaxée, on obtient pour $\omega$ fixé égal à $4/2$ les valeurs suivantes du rayon spectral de l'opérateur $M_h^{2h}$ :

| | $\nu = 1$ | $\nu = 2$ | $\nu = 3$ | $\nu = 4$ |
|---|---|---|---|---|
| $\rho(M_h^{2h})$ | 1/2 | 1/4 | 1/8 | 0.083 ... |

Dans le paragraphe suivant, nous allons étendre la méthode multigrille au cas bidimensionnel en résolvant l'équation de Poisson avec les conditions aux limites de Dirichlet sur le carré unité.

## 3.3 METHODE MULTIGRILLE BIDIMENSIONNELLE.

### 3.1. Problème modèle

Soit à résoudre l'équation de Poisson

$$-\Delta u = -\frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial x^2} = f \quad \text{sur} \quad \Omega = (0,1)\times(0,1) \qquad (3.12)$$

avec les conditions aux limites de Dirichlet

$$u = g \qquad \text{sur} \quad \Gamma = \partial\Omega$$

### 3.2. Formulation discrète du problème

Soit $\Omega_h = \left\{x = (ih, jh) : i,j = 1,\ldots, N_h-1\right\}$ où $N_h = \frac{1}{h}$
et soit $\Omega_H = \Omega_{2h} = \left\{x = (2ih, 2jh) : i,j = 1,\ldots, \frac{N_h}{2} - 1\right\}$

Si l'on discrétise le problème (3.12) par la méthode des différences finies sur $\Omega_h$, on obtient en utilisant la notation moléculaire l'approximation du second ordre classique :

$$L_h^{\Omega} = -\Delta_h = \frac{1}{h^2}\begin{bmatrix} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{bmatrix}$$

L'élimination des conditions aux limites discrètes conduit à une équation de grille de la forme :

$$L_h u_h := \frac{1}{h^2}\begin{bmatrix} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{bmatrix} u_h = f_h \qquad (\Omega_h) \qquad (3.13)$$

où $f_h$ contient une composante de la frontière $\Gamma$ en tous les points $x = (ih, jh)$ avec $i$ ou $j = 1$ ou $N_h-1$

Observons que la représentation matricielle du système(3.13) dépend de l'ordre de parcours des points de la grille $\Omega_h$. En effet, si nous choisissons l'ordre lexicographique c'est-à-dire si les points de la grille sont indicés comme l'indique la figure 1 ,

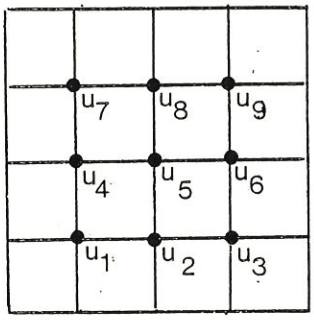



fig.1

Nous obtenons la représentation matricielle de $L_h$ suivante :

$$L_h = \frac{1}{h^2} \begin{bmatrix} 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 4 & 0 & 0 & -1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 4 & -1 & 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & -1 & 4 & -1 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & -1 & 4 & 0 & 0 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 4 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 \end{bmatrix}$$

Par contre, en choisissant l'ordre "red-black" c'est-à-dire en indiçant les points comme sur la figure ci-dessous

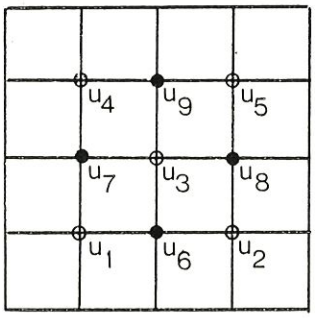



fig.2

o point rouge

● point noir

— Nous obtenons pour $L_h$ la représentation matricielle :

$$L_h = \frac{1}{h^2}\left[\begin{array}{ccccc|cccc} 4 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 & -1 & 0 & -1 & 0 \\ 0 & 0 & 4 & 0 & 0 & -1 & -1 & -1 & -1 \\ 0 & 0 & 0 & 4 & 0 & 0 & -1 & 0 & -1 \\ 0 & 0 & 0 & 0 & 4 & 0 & 0 & -1 & -1 \\ \hline -1 & -1 & -1 & 0 & 0 & 4 & 0 & 0 & 0 \\ -1 & 0 & -1 & -1 & 0 & 0 & 4 & 0 & 0 \\ 0 & -1 & -1 & 0 & -1 & 0 & 0 & 4 & 0 \\ 0 & 0 & -1 & -1 & -1 & 0 & 0 & 0 & 4 \end{array}\right]$$

Dans ce paragraphe, nous allons traiter le cas où $L_h$ est la matrice associée à l'ordre lexicographique, tous les développements ultérieurs pouvant être traités de manière similaire pour le cas de l'ordre "red-black" Observons tout d'abord que $L_h$ est une matrice symétrique dont les vecteurs propres sont donnés par :

$$\varphi^n(x) = 2 \sin(n_1 \pi x_1) \sin(n_2 \pi x_2) \qquad x \in \Omega_h ; |n| \leqslant N_h - 1$$

où $n = (n_1, n_2) \in \mathbb{N}^2$

et $|n| = \max(n_1, n_2)$

Ce qui donne dans le cas particulier où $N_h = 4$ :

$$\varphi^n(x) = \begin{pmatrix} \sin(n_1\pi h)\sin(n_2\pi h) ; \sin(2n_1\pi h)\sin(n_2\pi h) ; \sin(3n_1\pi h)\sin(n_2\pi h) \\ \sin(n_1\pi h)\sin(2n_2\pi h) ; \sin(2n_1\pi h)\sin(2n_2\pi h) ; \sin(3n_1\pi h)\sin(2n_2\pi h) \\ \sin(n_1\pi h)\sin(3n_2\pi h) ; \sin(2n_1\pi h)\sin(3n_2\pi h) ; \sin(3n_1\pi h)\sin(3n_2\pi h) \end{pmatrix}$$

Les valeurs propres correspondantes de $L_h$ sont toutes <u>positives</u> et données par l'expression :

$$K_n = \frac{1}{h^2} \left(2 - \cos(n_1\pi h) - \cos(n_2\pi h)\right)$$

### 3.3.3. Méthode deux-grilles

Comme dans le cas unidimensionnel, nous allons considérer deux phases : le lissage et la correction par grille grossière.

#### 1- Le lissage

Un pas de la méthode de Jacobi relaxéeappliquée au problème(3.12)avec $w_h$ comme première approximation sera noté :

$$\bar{w}_h = \text{RELAX}\ (\ w_h, L_h, f_h; \omega\ )$$

Où la matrice d'itération associée est donnée par l'expression

$$S_h = S_h(\omega) = I_h - \frac{\omega h^2}{4} L_h$$

Par cette formulation, on constate que les vecteurs propres de $L_h$ et ceux de $S_h(\omega)$ sont les mêmes :

$$\varphi^n(x) = 2 \sin(n_1 \pi x_1) \sin(n_2 \pi x_2) \qquad x \in \Omega_h\ ,\ |n| \leq N_h - 1$$

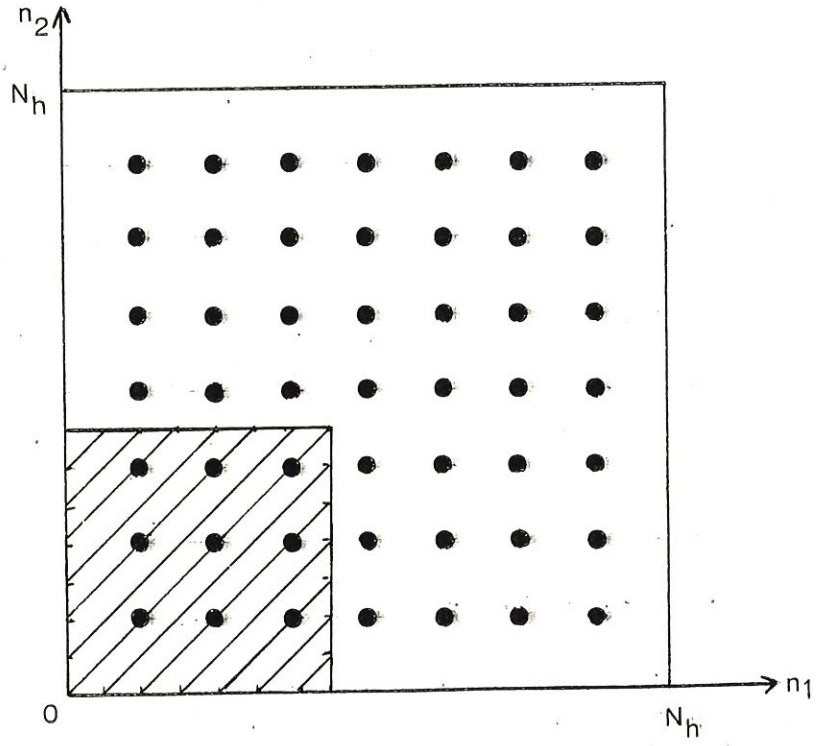
Les valeurs propres de $S_h(\omega)$ sont données par :

$$\chi_n = 1 - \frac{\omega}{2} (\ 2 - \cos(n_1 \pi h) - \cos(n_2 \pi h)) \qquad (3.14)$$

Comme pour le cas unidimensionnel, les propriétés de lissage de sont mesurées en distinguant basses et hautes fréquences (par rapport à la grille grossière $\Omega_H$ choisie).

Rappelons que les basses fréquences sont les vecteurs propres représentables sur la grille grossière $\Omega_H$ tandis que les vecteurs haute fréquence ne sont "pas visibles" sur $\Omega_H$

On définira en conséquence les vecteurs propres basse fréquence comme étant les $\varphi^n$ pour lesquels $|n| < N_h/2$ (dans la figure ci-dessous, ils occupent la partie hachurée). Les vecteurs propres haute fréquence seront les $\varphi^n$ pour lesquels $N_h/2 \leqslant |n| \leqslant N_h - 1$ (ils se situent dans la surface non hachurée)




Comme pour le cas unidimensionnel, nous définissons le facteur de lissage $\mu_h(\omega)$ de l'opérateur $S_h(\omega)$ par l'expression :

$$\mu_h(\omega) := \max\left\{ |\chi_n| : N_h \leqslant |n| \leqslant N_h - 1 \right\} \qquad (3.15)$$

Ou encore, le facteur de lissage est le plus mauvais ( ie le plus grand) facteur de lissage des composantes de l'erreur par pas d'itération de la méthode de relaxation employée.

En appliquant la définition(3.15) pour les valeurs trouvées en(3.14), il vient :

$$\mu(h,\omega) := \max \left\{ |1 - \omega(2-\cos\pi h)/2|, |1 - \omega(1+\cos\pi h)| \right\} \qquad (3.16)$$

et nous obtenons :

$$\begin{cases} \mu(h,\omega) \geqslant 1 & \text{pour } \omega \leqslant 0 \text{ ou } \omega > 1 \\ \mu(h,\omega) < 1 & \text{pour } 0 < \omega < 1 \end{cases} \qquad \text{(avec h petit)}$$

En particulier, l'expression(3.16) donne :

$$\mu(h,\omega) = \begin{cases} \cos\pi h & \text{si } \omega = 1, \\ (2 + \cos\pi h)/4 & \text{si } \omega = 1/2, \\ (1 + 2\cos\pi h)/5 & \text{si } \omega = 4/5. \end{cases}$$

Le suprémum de $\mu(h,\omega)$ par rapport à h sera noté $\mu^*(\omega)$ avec :

$$\mu^*(\omega) := \sup \left\{ \mu(h,\omega) : h < 1/4 \right\} \qquad (3.17)$$

La forme(3.16) donne :

$$\mu^*(\omega) = \max \left\{ |1 - \omega/2| , |1 - 2\omega| \right\}$$

$$\begin{cases} 1 - \omega/2 & \text{si } \omega \leqslant 4/5 \\ 1 - 2\omega & \text{si } \omega > 4/5 \end{cases}$$

Cette fonction rest représentée dans la figure ci-dessous :

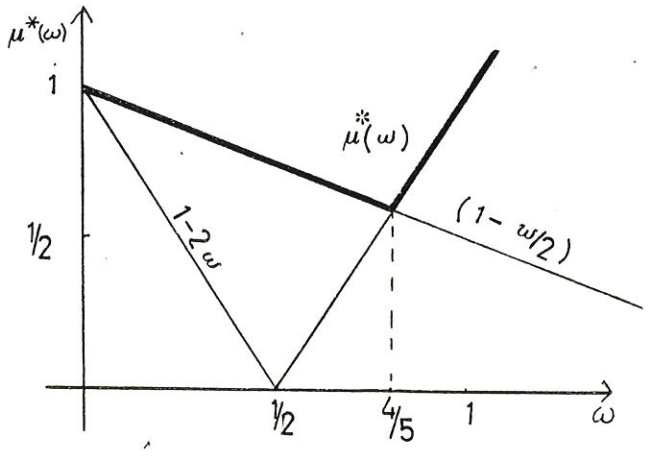



fig. 3

Nous obtenons en particulier

$$\mu^*(\omega) = \begin{cases} 1 & \text{si } \omega = 1 \\ 3/4 & \text{si } \omega = 1/2 \\ 3/5 & \text{si } \omega = 4/5 \end{cases}$$

La valeur optimale de $\mu^*(\omega)$ est donnée par $\omega = 4/5$ (cfr figure 3 )

2- Correction par grille grossière

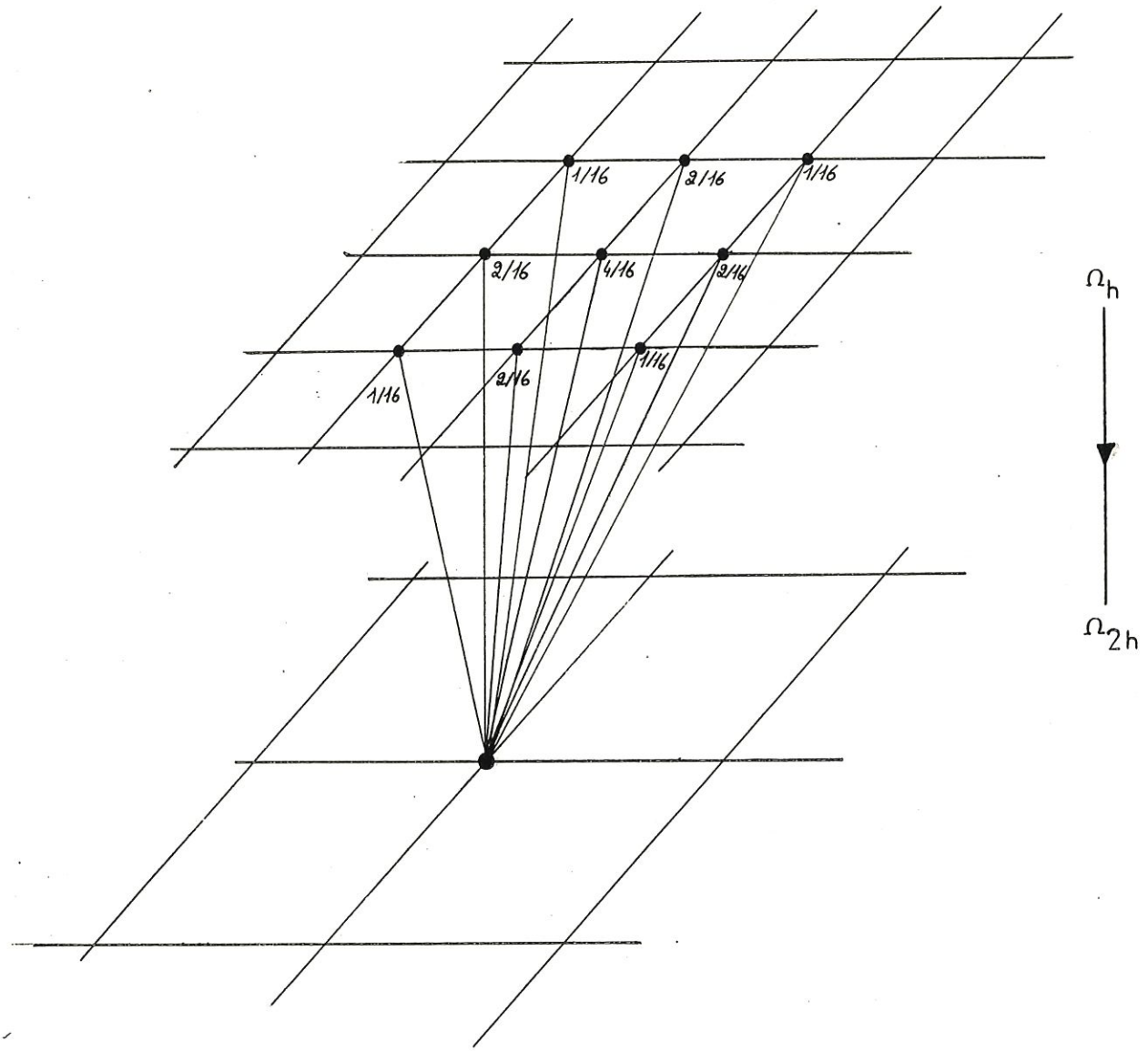
Rappelons que le passage de l'équation du défaut $L_h v_h^j = \bar{d}_h^j$ sur $\Omega_h$ à l'équation $L_H \hat{v}_H^j = \bar{d}_H^j$ sur $\Omega_H$ nécessite deux opérateurs de transferts : un opérateur de restriction $I_h^H$ qui envoie le défaut $\bar{d}_h^j$ sur sa restriction $\bar{d}_H^j$ et un opérateur d'interpolation $I_H^h$ qui envoie la correction $\hat{v}_H^j$ sur sa prolongée $\hat{v}_h^j$

L'opérateur de restriction : $I_h^H$

Nous utiliserons pour opérateur de restriction l'opérateur d'interpolation pondérée, en notation moléculaire :

$$I_h^{2h} := \frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Ce qui est encore illustré dans la figure suivante :




L'opérateur d'interpolation : $I_H^h$

Nous choisissons comme opérateur de prolongement l'opérateur d'interpolation bilinéaire, c'est-à-dire en notation moléculaire :

$$I_{2h}^{h} := \frac{1}{4} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 1 \\ 1 & 2 & 1 \end{bmatrix}$$

## 3- Définition de l'opérateur deux-grilles

### Description d'une itération deux-grilles :

Pour résoudre l'équation de Poisson discrétisée sur le carré unité, nous considérons l'algorithme décrit page 45. La $j^{ième}$ itération calculant $u_h^{j+1}$ à partir de $u_h^j$ se décompose comme suit :

<u>lissage</u> : calcul de $\bar{u}_h^j$ en appliquant $\nu_1$ pas de la méthode de Jacobi avec un facteur de relaxation $\omega$ à $u_h^j$ :

$$\bar{u}_h^j = \text{RELAX}^{\nu_1}(u_h^j, L_h, f_h; \omega)$$

<u>correction par grille grossière</u> :

- calcul du défaut

$$\bar{d}_h^j := f_h - L_h \bar{u}_h^j$$

- restriction du défaut

$$\bar{d}_{2h}^j := I_h^{2h} \bar{d}_h^j$$

en utilisant pour $I_h^{2h}$ l'opérateur d'interpolation pondérée

$$I_h^{2h} = \frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

- calcul de $v_{2h}^j$ solution exacte de l'équation du défaut sur $\Omega_{2h}$ (ici $L_{2h}$ est défini de manière analogue à $L_h$ )

$$L_{2h} v_{2h}^j = \bar{d}_{2h}^j$$

- interpolation de la correction $\hat{v}_{2h}^{j}$ en utilisant l'opérateur d'interpolation bilinéaire $I_{2h}^{h}$ ,

$$I_{2h}^{h} := \frac{1}{4} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

- calcul de l'approximation corrigée sur $\Omega_h$ :

$$\bar{u}_h^j + \hat{v}_h^j$$

lissage : calcul de $u_h^{j+1}$ en appliquant $\nu_2$ pas de la méthode de Jacobi avec un facteur de relaxation $\omega$ à $\bar{u}_h^j + \hat{v}_h^j$ :

$$\bar{u}_h^{j+1} := \text{RELAX}^{\nu_2} (\bar{u}_h^j + \hat{v}_h^j , L_h , f_h ; \omega )$$

Rayon spectral de l'opérateur deux-grilles

Nous considérons l'opérateur deux-grilles

$$M_h^{2h} = S^{\nu_2}(\omega)\ K_h^{2h}\ S^{\nu_1}(\omega)$$

$$= S^{\nu_2}(\omega)\ (I_h - I_{2h}^{h} L_{2h}^{-1} I_h^{2h}\ L_h)\ S^{\nu_1}(\omega)$$

dont les composantes ont été définies ci-dessus.

Comme dans le cas unidimensionnel, nous allons mettre en évidence $\hat{K}_h^{2h}$ équivalent orthogonal de $K_h^{2h}$ . Où $\hat{K}_h^{2h}$ est diagonale par bloc, composée de blocs $\hat{K}_{h,n}^{2h}$ de dimension 4 x 4 (au plus).

Pour cela, nous allons montrer que les espaces $E_{h,n}$ définis comme :

$$E_{h,n} := \text{span} \left\{ \varphi^{n_1 , n_2}, \varphi^{N_h - n_1 , N_h - n_2}, \varphi^{N_h - n_1 , n_2}, \varphi^{n_1 , N_h - n_2} \right\}$$

$( \text{Inl} \leqslant N_h/2 )$

sont laissés invariants par l'opérateur $K_h^{2h}$ .

Pour la description des matrices $\hat{K}^{2h}_{h,n}$ nous utiliserons aussi une base de vecteurs propres de $L_{2h}$ donnée par :

$$\Phi^n(x) := 2\sin(n_1\pi x)\sin(n_2\pi x_2)$$

$$\text{pour } x\in\Omega_{2h} \; ; |n| \leqslant N_h/2 - 1$$

Sur la grille grossière $\Omega_{2h}$, nous avons les égalités :

$$\Phi^{n_1,n_2}(x) = \varphi^{n_1,n_2}(x) = \varphi^{N_h-n_1,N_h-n_2}(x) = -\varphi^{N_h-n_1,n_2}(x) = -\varphi^{n_1,N_h-n_2}(x)$$

$$\text{pour } x\in\Omega_{2h} \; ; |n| \leqslant N_h/2 - 1$$

Pour $n_1 = N_h/2$ et/ou $n_2 = N_h/2$, les espaces $E_{h,n}$ sont respectivement de dimension 1 ou 2 et leurs vecteurs propres de base coïncident sur $\Omega_{2h}$ avec la fonction de grille nulle.

Les opérateurs composant $K^{2h}_h$ possèdent les propriétés :

$\underline{L_h : E_{h,n} \longrightarrow E_{h,n}}$ : car les vecteurs générant $E_{h,n}$ sont aussi vecteurs propres de $L_h$.

$\underline{I^{2h}_h : E_{h,n} \longrightarrow \text{span}\{\Phi^n\}}$ pour $n \leqslant N_h/2 - 1$

$$I^{2h}_h \begin{pmatrix} \varphi^{n_1,n_2} \\ \varphi^{N_h-n_1,N_h-n_2} \\ -\varphi^{N_h-n_1,n_2} \\ -\varphi^{n_1,N_h-n_2} \end{pmatrix} = \begin{pmatrix} (1-\xi)(1-\eta) \\ \xi\eta \\ \xi(1-\eta) \\ (1-\xi)\eta \end{pmatrix} \Phi^{n_1,n_2}$$

pour $|n| \leqslant N_h/2 - 1$ et où $\begin{cases} \xi = \sin^2(n_1\pi/2N_h), \\ \eta = \sin^2(n_2\pi/2N_h) \end{cases}$

$$I^{2h}_h \varphi^{n_1,n_2} = 0 \quad \text{pour } n_1 = N_h/2 \text{ et/ou } n_2 = N_h/2$$

$\underline{L_{2h} : \text{span}\{\Phi^n\} \longrightarrow \text{span}\{\Phi^n\}}$ : car les vecteurs $\Phi^n$ sont vecteurs propres de $L_{2h}$.

$$\underline{\underline{I_{2h}^{h} : \text{span}\{\Phi^{n}\} \longrightarrow E_{h,n}}} \quad \text{pour} \quad |n| \leq N_h/2 - 1$$

$$I_{2h}^{h}\ \Phi^{n_1,n_2} = (1-\xi)(1-\eta)\,\varphi^{n_1,n_2} + \xi\eta\,\varphi^{N_h-n_1,N_h-n_2}$$

$$= \xi(1-\eta)\,\varphi^{N_h-n_1,n_2} - (1-\xi)\eta\,\varphi^{n_1,N_h-n_2}$$

Nous pouvons ainsi décrire les blocs composant la matrice $\hat{K}_h^{2h}$ :

$$\hat{K}_{h,n}^{2h} = \begin{cases} \cdot\ I - \frac{1}{\Lambda}(b_i c_j)_{4,4} & |n| < N_h/2 \\ \cdot\ \text{matrice d'identité de dimension 2} & \text{si } n_1 \text{ ou } n = N_h/2 \\ \cdot\ \text{matrice d'identité de dimension 1} & \text{si } n_1 = n_2 = N_h/2 \end{cases}$$

où $\Lambda = \xi(1-\xi) + \eta(1-\eta)$ et

$$b_1 = (1-\xi)(1-\eta), b_2 = \xi\eta, \quad b_3 = \xi(1-\eta), \quad b_4 = (1-\xi)\eta$$

$$c_1 = b_1(\xi+\eta), \quad c_2 = b_2(2-\xi-\eta), c_3 = b_3(1-\xi+\eta), c_4 = b_4(1+\xi-\eta)$$

Nous avons aussi :

$$\underline{\underline{S_h(\omega) : E_{h,n} \longrightarrow E_{h,n}}}$$ car nous avons vu que les vecteurs $\varphi^n$ sont aussi vecteurs propres de $S_h(\omega)$.

Par (3.14), nous pouvons décrire les blocs composant la matrice diagonale $\hat{S}_h(\omega)$, équivalent orthogonal de $S_h(\omega)$ comme :

$$\hat{S}_{h,n}(\omega) = \begin{cases} \begin{bmatrix} (1-\omega(\xi+\eta)) & & & \\ & (1-\omega(2-\xi-\eta)) & & \\ & & (1-\omega(1-\xi+\eta)) & \\ & & & (1-\omega(1+\xi-\eta)) \end{bmatrix}_{4,4} & \text{si } |n| < N_h/2 \\ \begin{bmatrix} 1-\omega(\xi+\eta) & \\ & 1-\omega(2-\xi-\eta) \end{bmatrix}_{2,2} & \text{si } n_1 \text{ ou } n_2 = N_h/2 \\ \begin{bmatrix} 1-\omega(\xi+\eta) \end{bmatrix}_{1,1} & \text{si } n_1 = n_2 = N_h/2 \end{cases}$$

Les espaces $E_{h,n}$ ($|n| \leqslant N_h/2$) étant laissés invariant par les opérateurs $K_h^{2h}$ et $S_h(\omega)$ , ils le sont aussi par $M_h^{2h}$ .

Nous obtenons la matrice diagonale par bloc $\hat{M}_h^{2h}$ , équivalent orthogonal de $M_h^{2h}$ . Avec :

$$\hat{M}_{h,n}^{2h} = \hat{M}_{h,n}^{2h}(\nu_1,\nu_2;\omega) = \hat{S}_{h,n}^{\nu_2}(\omega)\ \hat{K}_{h,n}^{2h}\ \hat{S}_{h,n}^{\nu_1}(\omega)$$

Nous avons donc réduit le calcul du rayon spectral de la matrice d'itération de la méthode deux-grilles ($\rho(M_h^{2h})$) au calcul des rayons spectraux de matrices au plus de dimension 4 :

$$\rho(M_h^{2h}) = \max\left\{ \rho(\hat{M}_{h,n}^{2h}) : |n| \leqslant N_h/2 \right\}$$

Comme dans le cas unidimensionnel, ce rayon spectral dépend de $\nu = \nu_1 + \nu_2$ , du paramètre $\omega$ et du pas de discrétisation $h$ .

Il sera encore noté :

$$\rho(h,\nu;\omega) := \rho(M_h^{2h}(\nu_1,\nu_2;\omega))$$

Nous introduisons encore le facteur $\rho^*(\nu,\omega)$ défini comme :

$$\rho^*(\nu,\omega) = \sup\left\{ \rho(h,\nu;\omega) : h \leqslant 1/4 \right\}$$

Dans le tableau ($I$) proposé par Trottenberg [11] , nous trouvons quelques valeurs de $\rho^*$ donné comme fonction de $\nu$ et de deux paramètres de relaxation fixés $\omega = 0,5$ et $\omega = 0,8$.

| $\nu$ | $\omega = 0,5$ | $\omega = 0,8$ |
|---|---|---|
| 1 | 0.750 | 0.600 |
| 2 | 0.563 | 0.360 |
| 3 | 0.422 | 0.216 |
| 4 | 0.316 | 0.137 |
| 5 | 0.237 | 0.113 |

Tableau ($I$) : comparaison des facteurs de convergence $\rho^*$ pour différentes valeurs de $\nu$ et de $\omega$ .

Nous constatons que les facteurs de convergence $\rho^*$ diminuent avec l'augmentation des valeurs de $\nu$ . Observons qussi que le facteur de relaxation $\omega$ = 0.8 donne de meilleurs facteurs de convergence que la valeur $\omega$ = 0.5 (ce qui avait déjà été suggéré par la valeur des facteurs de lissage respectifs).

Donnons, pour terminer ce chapitre, quelques valeurs de $\rho(h,\nu;\omega)$ comme une fonction de h dans le tableau (II) (Trottenberg [11] )

| h | $\omega$ = 0.5 | | | | $\omega$ = 0.8 | | | |
|---|---|---|---|---|---|---|---|---|
| | $\nu$ = 1 | $\nu$ = 2 | $\nu$ = 3 | $\nu$ = 4 | $\nu$ = 1 | $\nu$ = 2 | $\nu$ = 3 | $\nu$ = 4 |
| 1/4 | 0.667 | 0.458 | 0.310 | 0.217 | 0.483 | 0.233 | 0.171 | 0.130 |
| 1/8 | 0.731 | 0.534 | 0.391 | 0.285 | 0.570 | 0.324 | 0.185 | 0.130 |
| 1/16 | 0.745 | 0.555 | 0.414 | 0.308 | 0.592 | 0.351 | 0.208 | 0.135 |
| 1/32 | 0.749 | 0.561 | 0.420 | 0.314 | 0.598 | 0.358 | 0.214 | 0.137 |
| 1/64 | 0.750 | 0.562 | 0.421 | 0.316 | 0.600 | 0.359 | 0.215 | 0.137 |
| 1/128 | 0.750 | 0.562 | 0.422 | 0.316 | 0.600 | 0.360 | 0.216 | 0.137 |
| $\rho^*(\nu;\omega)$ | 0.750 | 0.563 | 0.422 | 0.316 | 0.600 | 0.360 | 0.216 | 0.137 |

Tableau (II) : $(h,\nu;\omega)$ facteur de convergence de la méthode deux-grilles.

Ce tableau montre que le facteur $\rho$ tend vers $\rho^*$ assez rapidement Donc, dans les cas considérés, $\rho^*$ contient l'information essentielle sur la convergence de la méthode deux-grilles.

## 3.4. APPLICATIONS NUMERIQUES.

### 3.4.1. Résolution de l'équation de Poisson

La méthode multigrille appliquée à la résolution de l'équation de Poisson sur le carré unité a été programmée par FOERSTER et WHITSCH [1] et [11] conformément à l'organigramme de la page 45

Nous allons décrire les principales options du programme à savoir :

- le choix d'une première approximation
- la méthode de lissage
- le choix des grilles grossières et leur nombre
- la méthode de résolution directe utilisée sur la grille la plus fine
- les opérateurs de transfert (restriction et prolongement)

- le choix d'une première approximation

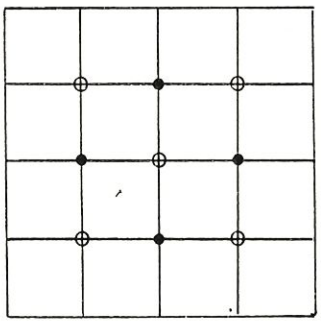
On adopte sur ce point une stratégie très générale :

$u_h = 0$ en tout point de $\Omega_h$

$u_h = f_h$ en tout point de $\Gamma_h$

- méthode de lissage

Notre choix se porte sur un opérateur de lissage "Red-Black" c'est-à-dire où un lissage complet se décompose en deux étapes correspondant respectivement à des opérations sur le sous ensemble des points rouges puis sur les points noirs de le grille $\Omega_h$ (voir figure)



o point rouge

• point noir

Dans la première étape, tous les points rouges de la grille sont traités par la méthode de Jacobi. Les valeurs calculées en ces points n'interfèrent pas et ne font intervenir que de l'information provenant des points noirs.

Cette étape terminée, on balaye alors de manière analogue tous les points noirs de la grille. Cette méthode de lissage est décrite par deux opérateurs partiels notés $S_h^{red}$ et $S_h^{black}$ respectivement et définis comme :

lissage partiel sur les points rouges

$$(S_h^{red}\,\zeta)(x) = \begin{cases} (I_h - \frac{h^2}{4} L_h)\,\zeta(x) & \text{si } x \text{ est un point rouge} \\ \zeta(x) & \text{si } x \text{ est un point noir} \end{cases}$$

lissage partiel sur les points noirs

$$(S_h^{black}\,\zeta)(x) = \begin{cases} \zeta(x) & \text{si } x \text{ est un point rouge} \\ (I_h - \frac{h^2}{4} L_h)\,\zeta(x) & \text{si } x \text{ est un point noir} \end{cases}$$

- choix des grilles grossières

Le choix d'une grille $\Omega_H$ , moins fine que la grille initiale $\Omega_h$ , est déterminé par $H$ . Ici, nous choisirons $H$ de façon standard en posant $H = 2h$

Dans le programme, la grille la plus fine correspond à une valeur de $h = 1/64$, le nombre de grilles possibles est ainsi limité à 6

( $h = \frac{1}{64}$ , $\frac{1}{32}$ , $\frac{1}{16}$ , $\frac{1}{8}$ , $\frac{1}{4}$ , $\frac{1}{2}$ )

- méthode de résolution directe utilisée sur la grille la plus fine

Observons tout d'abord les dimensions des différentes grilles :

$$\# \Omega_5 : (63)^2 = 3969$$

$$\# \Omega_4 : (31)^2 = 961$$

$$\# \Omega_3 : (15)^2 = 225$$

$$\# \Omega_2 : (7)^2 = 49$$

$$\# \Omega : (3)^2 = 9$$

$$\# \Omega_0 : (1)^2 = 1$$

On constate en particulier que la grille la plus grossière ne contient qu'un seul point intérieur, rendant la méthode de résolution d'un système linéaire évidente.

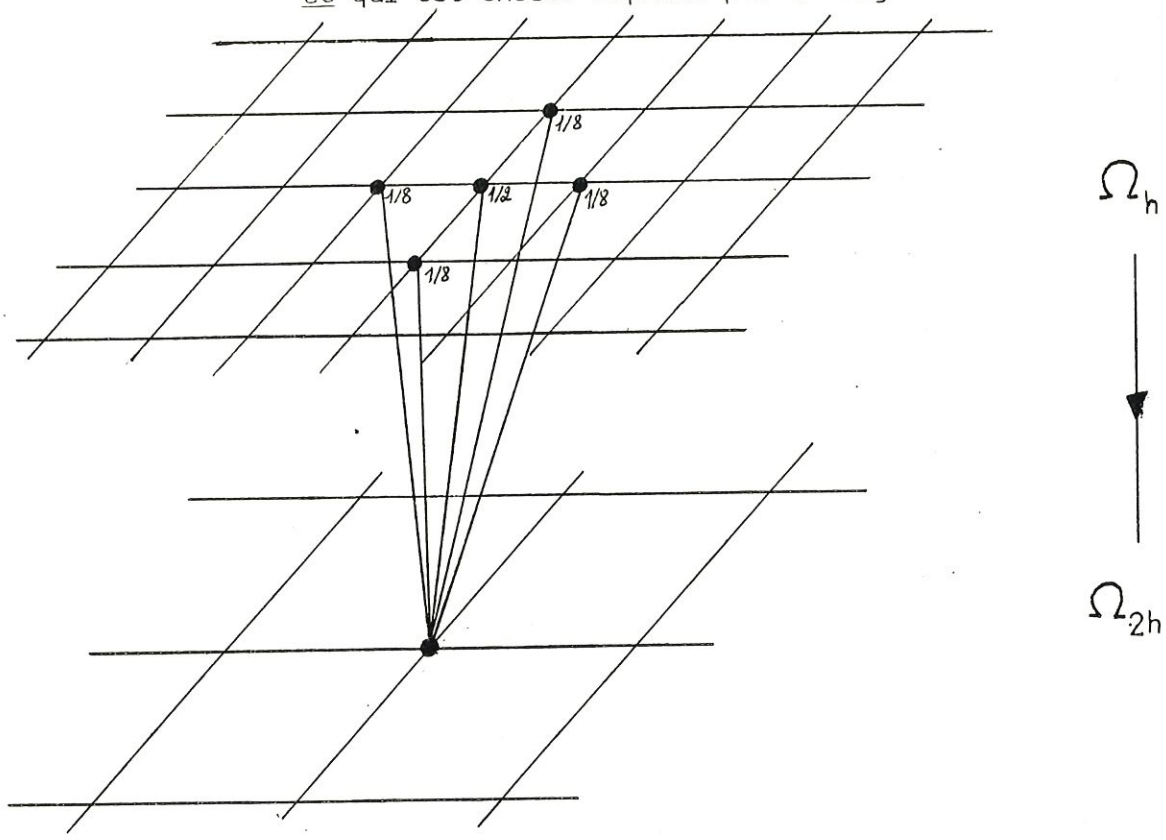
- les opérateurs de transfert

. l'opérateur de restriction : $I_h^{2h}$

Nous choisissons l'opérateur de restriction défini comme :

$$I_h^{2h} := \frac{1}{8} \begin{bmatrix} & 1 & \\ 1 & 4 & 1 \\ & 1 & \end{bmatrix}$$

ce qui est encore exprimé par la figure ci-dessous




Remarquons qu'après un pas de lissage, le défaut est nul sur tous les points noirs. Cela résulte immédiatement de la définition de l'opérateur de Laplace discret et de la stratégie "Red-Black".

La procédure de transfert de la grille fine vers la grille grossière devient alors extrêmement simple en particulier, pour le choix que nous avons fait de l'opérateur de restriction. Le transfert s'effectue en multipliant les points appartenant à l'intersection de la grille fine et de la grille grossière par un facteur 0.5 : Cet opérateur est appelé "demi-injection".

. l'opérateur de prolongement : $I_{2h}^{h}$

Nous choisissons un opérateur d'interpolation bilinéaire que que nous avons déjà évoqué dans ce travail :

$$I_{2h}^{h} = \frac{1}{4} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Nous pourrons cependant économiser des calculs si l'on observe la chose suivante. Après avoir effectué le transfert grille grossière vers grille fine, nous allons lisser la nouvelle approximation à l'aide de l'opérateur "Red-Black".
Le lissage sur les points rouges donne :

$$S_h^{red}\,\zeta(x) = \left( I_h - \frac{h^2}{4} L_h \right) \zeta(x)$$

$$= \left( \begin{bmatrix} & 0 & \\ 0 & 1 & 0 \\ & 0 & \end{bmatrix} - \begin{bmatrix} & -\frac{1}{4} & \\ -\frac{1}{4} & 1 & -\frac{1}{4} \\ & -\frac{1}{4} & \end{bmatrix} \right) \zeta(x)$$

$$= \begin{bmatrix} & \frac{1}{4} & \\ \frac{1}{4} & 0 & \frac{1}{4} \\ & \frac{1}{4} & \end{bmatrix} \zeta(x)$$

Ce qui signifie que le calcul des nouvelles valeurs de l'approximation sur les points rouges ne dépend que des valeurs sur les points noirs adjacents. Il sera donc inutile de procéder à l'interpolation des valeurs de la correction sur les points rouges de la grille fine.

Le code du programme FORTRAN écrit par FOERSTER et WITSCH [11] a été modifié pour permettre la résolution d'équations elliptiques plus générales. Nous allons traiter cette question dans le paragraphe suivant.

### 3.4.2. Résolution d'une équation elliptique non linéaire

Tout au long de ce travail, nous avons étudié des problèmes linéaires. Nous allons montrer qu'il est possible d'étendre le champ d'application de la méthode multigrille à des problèmes non linéaires.

Soit le problème :

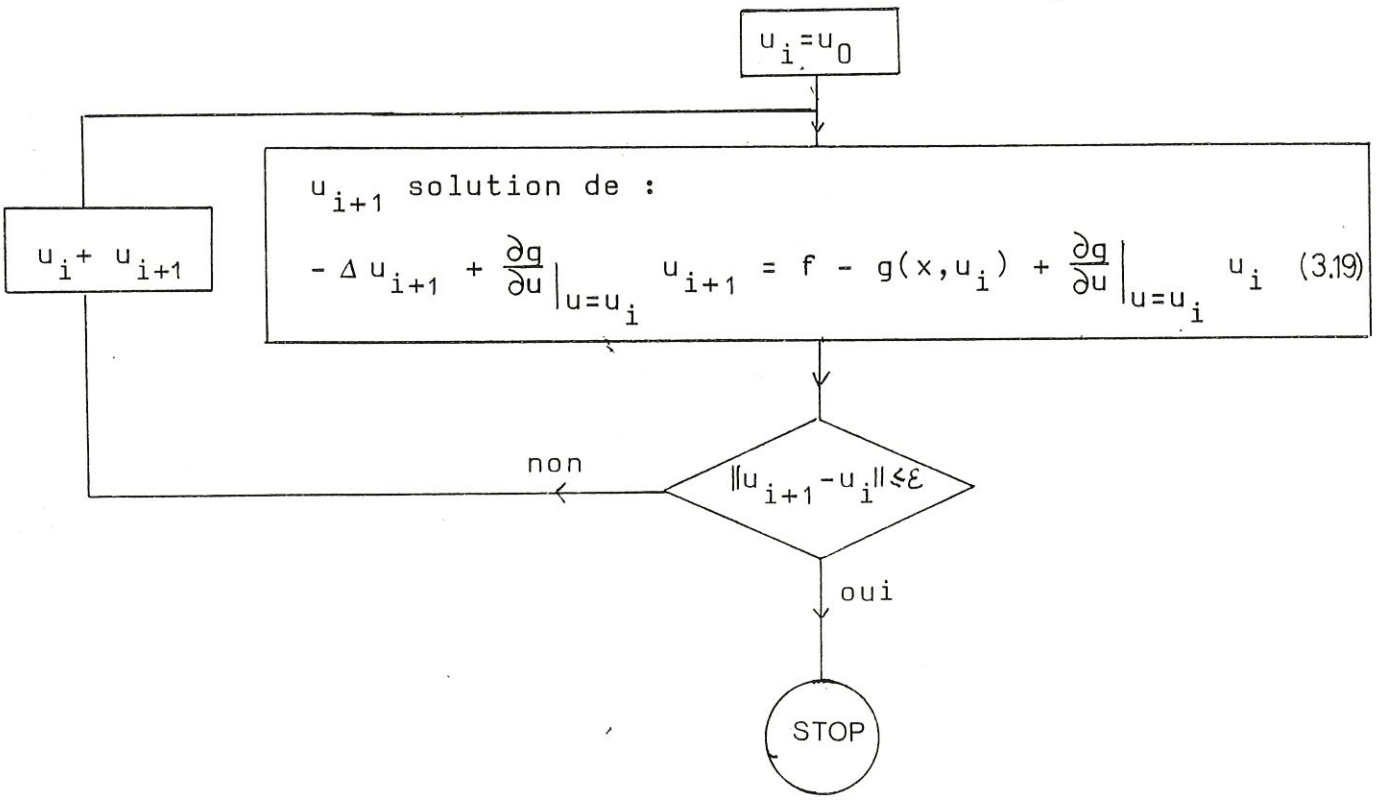
$$\begin{cases} \mathbb{L}^{\Omega}\, u(x) = f^{\Omega}(x) & x \in \Omega \\ \mathbb{L}^{\Gamma}\, u(x) = f^{\Gamma}(x) & x \in \Gamma := \partial\Omega \end{cases} \tag{3.18}$$

où

$L$ est cette fois un opérateur de la forme $-\Delta u + g^{\Omega}(x,u)$;

$g$ étant une fonction suffisamment régulière de $x$ et de $u$ .

Nous allons résoudre cette équation par la méthode de Newton appliquée au système linéarisé (cfr schéma ci-dessous)



A chaque étape (3.19 ), nous pouvons utiliser la méthode multigrille pour résoudre le système linéaire correspondant.

Remarquons que le système linéarisé est de la forme

$$- \Delta u + c\, u = f$$

donc un peu plus général que l'équation de Poisson traitée au § 3.4.1. L'extension de la méthode à ce type d'équation ne pose aucun problème.

Nous avons étendu le programme de calcul linéaire [1] , [11] à la résolution du système ( 3.18 ). Le code FORTRAN de ce programme est repris dans l'annexe 3.

Pour illustrer cette extension de la méthode au cas non linéaire nous allons résoudre le système :

$$\begin{cases} - \Delta u + \lambda u + u^3 = f & \text{dans } \Omega \\ u = 0 & \text{sur } \Gamma = \partial\Omega \end{cases} \qquad (3.20)$$

L'équation ( 3.20 ) a été étudiée par R. TEMAN [12] qui a démontré l'existence de l'unicité de la solution dans des conditions très générales.

Nous nous sommes placés dans les conditions suivantes :

$$\Omega = (0,1) \times (0,1)$$

$$f = -200\, x_1 (x_1 - 1) - 200\, x_2 (x_2 - 1) + x_1 (x_1 - 1) x_2 (x_2 - 1) + 10^6\, x_1^3 (x_1 - 1)^3 x_2^3 (x_2 - 1)^3$$

$$\lambda = 100$$

On peut montrer que la solution exacte est :

$$u = 100\, x_1 x_2 (x_1 - 1)(x_2 - 1) \quad .$$

Quatre itérations de la méthode de Newton suffisent pour atteindre une précision de l'ordre du millionième (voir annexe 3)

Aussi, en couplant les avantages de la méthode multigrille et ceux de la méthode de Newton, on peut s'attendre à d'excellents résultats dans la résolution de problèmes non linéaires.

# 4. CONCLUSIONS

Dans ce travail, nous avons présenté la méthode multigrille, une méthode de résolution de systèmes linéaires provenant d'équations aux dérivées partielles elliptiques linéaires.

Ces systèmes présentent les caractéristiques suivantes :

- une structure régulière
- une grande taille
- un grand nombre de composantes nulles.

Cette méthode est en fait la composée de méthodes classiques (Jacobi, Gauss-Seidel relaxées ou non, correction par grille grossière (C.G.C.), ...) dont elle a su tirer parti des propriétés les plus adéquates (lissage pour Jacobi ou Gauss-Seidel, écrêtage des composantes hautes fréquences de la solution pour C.G.C.).

Nous avons montré que la méthode multigrille était simple dans sa conception et que les paramètres qui la caractérisaient étaient faciles à quantifier.

Elle est en outre très flexible dans la mesure où ses composantes peuvent être choisies au mieux du problème considéré.

Sa programmation ne pose aucun problème et s'est avérée facile à modifier.

Enfin, les résultats comparatifs avec d'autres méthodes sont des plus encourageant et confirment les résultats théoriques. En particulier, le nombre de calculs qu'elle nécessite est proportionnel au nombre de points de la grille avec une constante de proportionnalité très raisonnable. Cette dernière propriété ajoutée au fait que la méthode multigrille peut, du moins formellement, être adaptée à des problèmes plus complexes en fait un outil particulièrement prometteur. C'est pourquoi les applications qui en font usage se multiplient aujourd'hui, tant dans le domaine des équations elliptiques que dans celui des équations paraboliques, linéaires ou non.

*

*  *

*

# ANNEXE 1.

## Démonstration de la proposition 1 établissant l'équivalence entre le problème de Poisson de (2.3) et la formulation variationnelle (2.4)

### Démontrons la condition nécessaire :

L'espace des fonctions utilisées sera :

$$V = H_0^1(\Omega) = \left\{ v \; L^2(\Omega), \frac{\partial v}{\partial x_i} \in L^2(\Omega), i=1,\ldots,n; v/_\Gamma = 0 \right\}$$

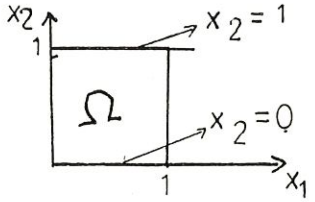
Prenons $h \in H_0^1(\Omega)$ , arbitraire, il vient

$$-\int\int_\Omega h(x).\Delta u(x)\, dx = \int\int_\Omega h(x).f(x)\, dx$$

$$-\int\int_\Omega h(x_1,x_2)\left(\frac{\partial^2 u}{\partial x_1^2}(x_1,x_2)+\frac{\partial^2 u}{\partial x_2^2}(x_1,x_2)\right) dx_1\, dx_2$$

$$= \int\int_\Omega h(x_1,x_2)\, f(x_1,x_2)\, dx_1\, dx_2 \quad (*)$$

Comme $\Omega$ a la forme particulière :




nous obtenons

$$\int\int_\Omega h(x_1,x_2)\frac{\partial^2 u}{\partial x_2^2}(x_1,x_2)\,dx_1\,dx_2 = \int_0^1 \left\{ \int_0^1 h(x_1,x_2)\frac{\partial^2 u}{\partial x_2^2}(x_1,x_2)\,dx_2 \right\} dx_1$$

Après intégration par parties, il vient

$$\int_0^1 \left\{ h(x_1,x_2)\frac{\partial u}{\partial x_2}(x_1,x_2)\Big|_{x_2=0}^{x_2=1} - \int_0^1 \frac{\partial h}{\partial x_2}(x_1,x_2)\,\frac{\partial u}{\partial x_2}(x_1,x_2)\,dx_2 \right\} dx_1$$

Le premier terme s'annule puisque $h \in H_0^1(\Omega)$

$$= -\int_0^1 \left\{ \int_0^1 \frac{\partial h}{\partial x_2}(x_1, x_2)\, \frac{\partial u}{\partial x_2}(x_1, x_2)\, dx_2 \right\} dx_1$$

$$= -\iint_\Omega \frac{\partial h}{\partial x_2}(x_1, x_2)\, \frac{\partial u}{\partial x_2}(x_1, x_2)\, dx_1\, dx_2$$

Ainsi (*) devient

$$\iint_\Omega \frac{\partial h}{\partial x_1}(x_1, x_2)\, \frac{\partial u}{\partial x_1}(x_1, x_2) + \frac{\partial h}{\partial x_2}(x_1, x_2)\, \frac{\partial u}{\partial x_2}(x_1, x_2)\, dx = \iint_\Omega f.h\, dx$$

D'où $\iint_\Omega \nabla h.\nabla u\, dx - \iint_\Omega f.h\, dx = 0 \quad \forall\, h \in H_0^1(\Omega)$

Ajoutons le terme positif $\frac{1}{2} \iint_\Omega |\nabla h|^2\, dx$, nous obtenons

$$\iint_\Omega \nabla h.\nabla u\, dx - \iint_\Omega f.h\, dx + \frac{1}{2} \iint_\Omega |\nabla h|^2\, dx \geqslant 0 \qquad \forall\, h \in H_0^1(\Omega)$$

ou encore $\iint_\Omega \frac{1}{2} \left[ |\nabla u|^2 + |\nabla h|^2 + 2\, \nabla u\, \nabla h \right] dx$ (**)

$$- \iint_\Omega f\,(h+u)\, dx - \frac{1}{2} \iint_\Omega |\nabla u|^2\, dx + \iint_\Omega f.v\, dx \geqslant 0 \quad \forall h \in H_0^1(\Omega)$$

Posons $J(v) = \frac{1}{2} \iint_\Omega |\nabla v|^2\, dx - \iint_\Omega f.v\, dx$

et $a(v,w) = \iint_\Omega \nabla v.\nabla w\, dx$

La relation (**) devient

$$\frac{1}{2} \iint_\Omega \left| \nabla(u+h) \right|^2 dx - \iint_\Omega f(u+h)\, dx - J(u) \geqslant 0 \qquad \forall\, h \in H_0^1(\Omega)$$

ou encore

$$J(u+h) - J(u) \geqslant 0 \qquad \forall\, h \in H_0^1(\Omega)$$

D'où

$$J(u) \leqslant J(u+h) \qquad \forall\, h \in H_0^1(\Omega)$$

et

$$J(u) = \operatorname{Inf}_{v \in H_0^1(\Omega)} J(v)$$

Ce qui démontre la condition nécessaire de la proposition et nous permet d'établir entre le problème de Dirichlet et le cas général la correspondance :

| Problème de Dirichlet | Cas général |
|---|---|
| $V = H_0^1(\Omega)$ | $V$ espace de Hilbert |
| $a(u,v) = \iint_\Omega \nabla u . \nabla v \, dx$ | $a(u,v)$ forme bilinéaire continue sur $V$ |
| $L(v) = \iint_\Omega f.u \, dx$ | $L(v)$ forme linéaire continue sur $V$ |
| $J(v)=\frac{1}{2}\iint_\Omega \lvert\nabla v\rvert^2 dx - \iint_\Omega f.v \, dx$ | $J(v) = \frac{1}{2} a(v,v) - L(v)$ |

Démontrons à présent la condition suffisante :

ai u est solution du problème d'optimisation (2.4)

alors u est solution du problème (2.3)

Partons de $J(v) = \frac{1}{2} \iint_\Omega \lvert\nabla v\rvert^2 \, dx - \iint_\Omega f.v \, dx$

avec $a(v,w) = \iint_\Omega \nabla v . \nabla w \, dx$

et $L(v) = \iint_\Omega f.v \, dx$

Il vient

$$J(u+\lambda v) = \frac{1}{2}(a(u,u)+\lambda^2 a(v,v)+2\lambda a(u,v)-2L(u)-2\lambda L(v))$$

où $u, v \in H_0^1(\Omega)$ et $\lambda \in \mathbb{R}$

Et

$$\frac{d}{d\lambda} J(u+\lambda v)\Big|_{\lambda=0} = 0$$

Calculons

$$\frac{d}{d\lambda}J(u+\lambda v)\Big|_{\lambda=0} = 2\ a(v,v) + 2a(u,v) - 2L(v)\Big|_{\lambda=0}$$

$$= 2a(u,v) - 2L(v)$$

d'où

$$a(u,v) = L(v) \qquad \forall\, v \in H_0^1(\Omega)$$

ou encore

$$\iint_\Omega \nabla u.\nabla v\ dx = \iint_\Omega f.v\ dx \qquad \forall\, v \in H_0^1(\Omega)$$

Et par la formule de Green :

$$-\iint_\Omega \Delta u.v\ dx + \iint_\Gamma v.\frac{\partial u}{\partial n}\ dx = \iint_\Omega f.v\ dx$$

le second terme s'annule puisque $v \in H_0^1(\Omega)$ , et nous obtenons

$$-\iint_\Omega \Delta u.v\ dx = \iint_\Omega f.v\ dx \qquad \forall\, v \in H_0^1(\Omega)$$

$$\iint_\Omega (\nabla u + f)v\ dx = 0 \qquad \forall\, v \in H_0^1(\Omega)$$

il vient

$\nabla u + f = 0$ : équation de Poisson

ce qui démontre la proposition et nous donne la correspondance

| Cas général | Problème de Dirichlet |
|---|---|
| $a(u,v) = L(v)$ $\forall v \in V$ | $\iint_\Omega \Delta u.v\ dx = -\iint_\Omega f.v\ dx$ $\forall\, v \in H_0^1(\Omega)$ |

# A N N E X E  2.

Programme de résolution d'un grand système linéaire, creux et non symétrique.

---

```
0001   C     ************************************************************
0002   C     LARGE SPARSE, UNSYMMETRIC SYSTEM OF LINEAR EQUATIONS SOLVER.
0003   C     ************************************************************
0004   C
0005   C     AUTHOR     : JOHN E.KEY(ALABAMA UNIVERSITY-HUNTSVILLE ALABAMA)
0006   C                  AUGUST 1972
0007   C     REVISED BY E. VERGISON (SOLVAY S.A.) SEPTEMBER 1981.
0008   C
0009   C     REFERENCE : COMPUTER PROGRAM FOR SOLUTION OF LARGE SPARSE
0010   C                 UNSYMMETRIC SYSTEMS OF LINEAR EQUATIONS.
0011   C                 JOHN E. KEY
0012   C                 NTIS - AD-755 878  - AUGUST 1972
0013   C
0014   C     SUBROUTINES CALLED : SASMX, SBSMX,... TO SISMX, CPUTIME (*)
0015   C                    (*)   CPUTIME IS A VAX 11/780 DEPENDENT
0016   C                          ROUTINE THAT TELLS THE USER THE CPU TIME
0017   C                          USED BY THE ALGORITHM; NEED TO BE ADAPTED
0018   C                          ON ANOTHER COMPUTER.
0019   C
0020   C     CALLING SEQUENCE
0021   C     ----------------
0022   C     CALL  SUSMX (NU,MR,MC,A,P,ICOL,ZTEST,ICC,IRC,JELE,JELIM,JCOL
0023   C                  NPIV,TIME)
0024   C
0025   C       NU     = NUMBER OF ROWS IN SYSTEM OF EQUATIONS TO BE SOLVED
0026   C       MR     = MAXIMUM NUMBER OF ROWS IN EQUATION SOLVER AND IN SYSTEM OF
0027   C                EQUATIONS TO BE SOLVED.
0028   C       MC     = MAXIMUM NUMBER OF COLUMNS IN EQUATION SOLVER
0029   C       A      = COEFFICIENT MATRIX IN COMPRESSED FORM.
0030   C       P      = VECTOR TO BE SOLVED.
0031   C       ICOL   = RECORDS POSITION OF NON-ZERO COEFF. IN SYSTEM OF EQUATIONS
0032   C                TO BE SOLVED
0033   C       ZTEST  = TEST TO DECIDE WHEN ELEMENTS OTHER THEN PIVOTAL MUST BE SET
0034   C                EQUAL TO ZERO.
0035   C       ICC    = NUMBER OF NON-ZERO COEFF. IN ANY GIVEN ROW OF A
0036   C       IRC    = NUMBER OF NON-ZERO COEFFICIENTS IN ANY GIVEN COLUMN OF A
0037   C       JELE   = NUMBER OF NON-ZERO COEFFICIENTS IN THE COEFFICIENT ARRAY A
0038   C       JELIM  = NUMBER OF TERMS ELIMINATED BY THE ELIMINATION PROCESS
0039   C       JCOL   = MAXIMUM NUMBER OF COLUMNS USED IN THE SOLUTION PROCESS
0040   C       NPIV   = PARAMETER DETERMINING THE PIVOT ALGORITHM
0041   C         NPIV   = 1 : GAUSS - JORDAN ELIMINATION ALGORITHM
0042   C         NPIV   = 2 : GAUSS - JORDAN PARTIAL PIVOTING ALGORITHM
```

```
0043    C          NPIV   = 3 : GAUSS - JORDAN FULL PIVOTING ALGORITHM
0044    C          NPIV   = 4 : MINIMUM ROW - MINIMUM COLUMN ALGORITHM
0045    C          NPIV   = 5 : MINIMUM COLUMN - MINIMUM ROW ALGORITHM
0046    C          NPIV   = 6 : MAXIMUM COLUMN - MINIMUM ROW ALGORITHM
0047    C          NPIV   = 7 : MINIMUM ROW ENTRIES TIMES COLUMN ENTRIES ALGORITHM
0048    C         TIME  = CPU TIME USED BY THE ALGORITHM
0049    C         A(I,1)= SOLUTION VECTORS RETURNED IN FIRST COLUMN OF A
0050    C
0051    C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
0052    C
0053          SUBROUTINE SOSMX (NU,MR,MC,A,P,ICOL,ZTEST,ICC,IRC,JELE,JELIM,JCOL
0054         1,                  NPIV,TIME)
0055    C
0056          REAL *8 A,C,P,X,ZTEST,ATEST
0057          INTEGER*4 PIVROW,PIVCOL,OPROW
0058          COMMON /SINTG1/ PIVROW,PIVCOL,OPROW
0059          COMMON /SINTG2/ IY
0060          COMMON /SINTG3/ LKJ
0061          DIMENSION IRC(MR),ICC(MR),A(MR,MC),ICOL(MR,MC),P(MR)
0062          IMP = 6
0063    C
0064    C     TELL THE USER WHICH PIVOTAL STRATEGY HE IS USING.
0065    C
0066          GOTO (610,620,630,640,650,660,670), NPIV
0067    610   WRITE(IMP,2100)
0068          GOTO 680
0069    620   WRITE(IMP,2200)
0070          GOTO 680
0071    630   WRITE(IMP,2300)
0072          GOTO 680
0073    640   WRITE(IMP,2400)
0074          GOTO 680
0075    650   WRITE(IMP,2500)
0076          GOTO 680
0077    660   WRITE(IMP,2600)
0078          GOTO 680
0079    670   WRITE(IMP,2700)
0080    680   CONTINUE
0081          TIME1 = 0
0082          CALL CPUTIME (TIME1)
0083    C
0084    C     INITIALIZE ROW ENTRY COUNTER
```

```
0085    C
0086            DO 99  I=1,NU
0087    99      IRC(I) = 0
0088    C
0089    C       COUNT ROW AND COLUMN ENTRIES
0090    C
0091            JELE = 0
0092            DO 101 I=1,NU
0093            DO 102 J=1,MC
0094            IC = ICOL(I,J)
0095            IF(IC.EQ.0) GOTO 101
0096            JELE = JELE+1
0097            ICC(I) = J
0098            IRC(IC) = IRC(IC)+1
0099    102     CONTINUE
0100    101     CONTINUE
0101            JCOL = 0
0102            JELIM = 0
0103            DO 800  I=1,NU
0104    800     IF(ICC(I).GT.JCOL) JCOL=ICC(I)
0105    C
0106    C       MAIN LOOP
0107    C
0108            DO 66  LKJ = 1,NU
0109    C
0110    C       CHOICE OF THE PIVOT ALGORITHM.
0111    C
0112            GOTO (710,720,730,740,750,760,770), NPIV
0113    C
0114    710     CALL SCSMX(IRC,ICC,ICOL,A,MR,MC,NU)
0115            GOTO 780
0116    720     CALL SDSMX(IRC,ICC,ICOL,A,MR,MC,NU)
0117            GOTO 780
0118    730     CALL SESMX(IRC,ICC,ICOL,A,MR,MC,NU)
0119            GOTO 780
0120    740     CALL SFSMX(IRC,ICC,ICOL,A,MR,MC,NU)
0121            GOTO 780
0122    750     CALL SGSMX(IRC,ICC,ICOL,A,MR,MC,NU)
0123            GOTO 780
0124    760     CALL SHSMX(IRC,ICC,ICOL,A,MR,MC,NU)
0125            GOTO 780
0126    770     CALL SISMX(IRC,ICC,ICOL,A,MR,MC,NU)
```

```
0127   780   CONTINUE
0128   C
0129   C     NORMALIZE PIVROW
0130   C
0131         X = A(PIVROW,IY)
0132         IC = ICC(PIVROW)
0133         DO 5  J=1,IC
0134   5     A(PIVROW,J) = A(PIVROW,J)/X
0135         A(PIVROW,IY) = 1.D+00
0136         P(PIVROW) = P(PIVROW)/X
0137   C
0138   C     SELECT ROWS THAT CAN BE OPERATED ON
0139   C
0140   C     SECOND LOOP
0141   C
0142         DO 106  I=1,NU
0143         IF(IRC(PIVCOL).EQ.1) GOTO 107
0144         IF(I.EQ.PIVROW) GOTO 106
0145         IC = IABS(ICC(I))
0146   C
0147   C     INNER LOOP
0148   C
0149         DO 105  J=1,IC
0150         IF(ICOL(I,J) - PIVCOL) 105,77,106
0151   C
0152   C     IF YOU CAN GET TO THIS POINT OPROW CONTAINS PIVOTAL ELEMENT
0153   C
0154   77    OPROW = I
0155         JKOP = 1
0156         JKPI = 1
0157         C = -A(OPROW,J)
0158         P(OPROW) = P(PIVROW)*C+P(OPROW)
0159   79    CONTINUE
0160         IF(ICOL(PIVROW,JKPI) .EQ.0) GOTO 106
0161         IF(ICOL(OPROW,JKOP).EQ.0) GOTO 80
0162         IF(ICOL(PIVROW,JKPI)-ICOL(OPROW,JKOP)) 80,81,82
0163   C
0164   C     OPROW DOES NOT CONTAIN THIS ELEMENT'ADD ELEMENT TO OPROW
0165   C
0166   80    ICC(I) = ICC(I)+1
0167         IF(ICC(I).LE.0) ICC(I) = ICC(I)-2
0168         II = IABS(ICC(I))
```

```
0169            IF(II.GT.JCOL) JCOL=II
0170            IF(II.LE.MC) GOTO 85
0171            WRITE(IMP,2000)
0172            RETURN
0173      85    JKL = JKOP+1
0174      90    IX = II-1
0175            A(OPROW,II) = A(OPROW,IX)
0176            ICOL(OPROW,II) = ICOL(OPROW,IX)
0177            II = IX
0178            IF(II.GE.JKL) GOTO 90
0179            A(OPROW,JKOP) = A(PIVROW,JKPI)*C
0180            ICOL(OPROW,JKOP) = ICOL(PIVROW,JKPI)
0181            IX = ICOL(OPROW,JKOP)
0182            IRC(IX) = IRC(IX)+1
0183            GOTO 83
0184      C
0185      C     PIVROW AND OPROW CONTAIN THIS ELEM. 'SHIFT BOTH AND OPERATE ON OPR
0186      C
0187      81    IX = ICOL(OPROW,JKOP)
0188            IF(IX.EQ.PIVCOL) GOTO 11
0189            X = A(PIVROW,JKPI)*C+A(OPROW,JKOP)
0190            A(OPROW,JKOP) = X
0191      C
0192      C     TEST OPROW TO SEE IF ANY ELEMENTS WERE ELIMINATED OTHER THAN
0193      C     THOSE IN THE PIVOTAL COLUMN
0194      C
0195            ATEST = DABS(X)-ZTEST
0196            IF(ATEST.GT.0.) GOTO 83
0197      11    IRC(IX) = IRC(IX)-1
0198            JELIM = JELIM+1
0199            ICC(OPROW) = ICC(OPROW)-1
0200            IF(ICC(OPROW)) 140,141,142
0201     141    CALL SASMX(IRC,ICC,ICOL,A,MR,MC,NU)
0202     140    CONTINUE
0203            ICC(OPROW) = ICC(OPROW)+2
0204     142    IX = IABS(ICC(OPROW))
0205            DO 131 NK = JKOP,IX
0206            A(I,NK) = A(I,NK+1)
0207     131    ICOL(I,NK) = ICOL(I,NK+1)
0208            IX = IX+1
0209            ICOL(I,IX) = 0
0210            JKPI = JKPI+1
```

```
0211            GOTO 79
0212    83      JKPI = JKPI+1
0213    C
0214    C       PIVROW DOES NOT CONTAIN THIS ELEMENT 'SHIFT OPROW AND CONTINUE
0215    C
0216    82      JKOP = JKOP+1
0217            GOTO 79
0218    105     CONTINUE
0219    106     CONTINUE
0220    107     CONTINUE
0221    C
0222    C       ELIMINATES PIVROW AND PIVCOL FROM BEING CONSIDERED AGAIN
0223    C
0224            ICC(PIVROW) = -ICC(PIVROW)
0225            IRC(PIVCOL) = -IRC(PIVCOL)
0226    86      CONTINUE
0227    C
0228    C       UNSCRAMBLE AND STORE SOLUTION IN FIRST COLUMN OF A
0229    C
0230            DO 350  I=1,NU
0231            II = ICOL(I,1)
0232    350     A(II,1) = P(I)
0233    C
0234    C       TIME ESTIMATION.
0235    C
0236            CALL CPUTIME (TIME)
0237            TIME = TIME-TIME1
0238            RETURN
0239    C
0240    C       PRINTING FORMATS.
0241    C
0242    2000    FORMAT(1H ,10X,'FIXED MATRIX DIMENSIONS EXCEEDED ')
0243    2100    FORMAT(1H0,10X,'GAUSS - JORDAN ELIMINATION ALGORITHM',//)
0244    2200    FORMAT(1H0,10X,'GAUSS - JORDAN PARTIAL PIVOTING ALGORITHM',//)
0245    2300    FORMAT(1H0,10X,'GAUSS - JORDAN FULL PIVOTING ALGORITHM',//)
0246    2400    FORMAT(1H0,10X,'MINIMUM ROW - MINIMUM COLUMN ALGORITHM',//)
0247    2500    FORMAT(1H0,10X,'MINIMUM COLUMN - MINIMUM ROW ALGORITHM',//)
0248    2600    FORMAT(1H0,10X,'MAXIMUM COLUMN - MINIMUM ROW ALGORITHM',//)
0249    2700    FORMAT(1H0,10X,'MINIMUM ROW ENTRIES TIMES COLUMN ',
0250           1 'ENTRIES ALGORITHM*',//)
0251            END
```

```
0001            SUBROUTINE SASMX(IRC,ICC,ICOL,A,MR,MC,NU)
0002     C      PRINTING IN CASE OF SINGULARITIES.
0003     C
0004            REAL*8 A
0005            INTEGER*4 PIVROW,PIVCOL,OPROW
0006            COMMON /SINTG1/ PIVROW,PIVCOL,OPROW
0007            COMMON /SINTG3/ LKJ
0008            DIMENSION IRC(MR),ICC(MR),ICOL(MR,MC),A(MR,MC)
0009            IB = 1
0010            DO 10  M=1,NU
0011            II = ICC(M)
0012     10     IF(II.GT.IB) IB=II
0013            WRITE(6,999) LKJ,PIVROW,PIVCOL,OPROW
0014            WRITE(6,801) (M,IRC(M),ICC(M),M=1,NU)
0015            WRITE(6,6)
0016     C
0017     C      PRINTING OF THE COEFFICIENT MATRIX.
0018     C
0019            CALL SBSMX(1,NU,IB,MR,MC,A,ICOL)
0020            WRITE(6,200)
0021     C
0022     C      PRINTING OF THE COLUMN INDEX MATRIX
0023            CALL SBSMX(2,NU,IB,MR,MC,A,ICOL)
0024            RETURN
0025     C
0026     C      PRINTING FORMATS
0027     C
0028     6      FORMAT('1 COEFFICIENT MATRIX')
0029     200    FORMAT('1 COLUMN IDENTIFICATION')
0030     801    FORMAT('0 COL/ROW NO.,' NO. COL. ENTRIES, NO. ROW ENTRIES',/
0031           1,(6(I7,2I5)))
0032     999    FORMAT('1   MATRIX SINGULAR',///,'   NO, CYCLES COMPLETED='
0033           1,I5,'  PIVROW=',I5,'  PIVCOL=',I5,'  OPROW=',I5)
0034            END
```

```
0001            SUBROUTINE SBSMX(IDUM,NR,NC,MR,MC,A,ICOL)
0002      C     PRINTING OF THE COEFFICIENT MATRIX (A) AND THE COLUMN INDEX
0003      C     MATRIX (ICOL) IN CASE OF SINGULARITIES.
0004      C
0005            REAL*8 A
0006            DIMENSION  A(MR,MC) ,ICOL(MR,MC)
0007            IPRINT = 12
0008            IF(IDUM.NE.1) IPRINT = 25
0009            IPR = IPRINT-1
0010            DO 35  K=1,NC,IPRINT
0011            MAX = K+IPR
0012            IF(MAX.GT.NC) MAX=NC
0013            IF(K.NE.1) WRITE(6,103)
0014            IF(IDUM.EQ.1) GOTO 45
0015            WRITE(6,101) (I,I=K,MAX)
0016            DO 20  J=1,NR
0017      20    WRITE(6,104) J,(ICOL(J,I),I=K,MAX)
0018            GOTO 35
0019      45    WRITE(6,102) (I,I=K,MAX)
0020            DO 40  J=1,NR
0021      40    WRITE(6,105) J,(A(J,I),I=K,MAX)
0022      35    CONTINUE
0023            RETURN
0024      C
0025      C     PRINTING FORMATS
0026      C
0027      101   FORMAT(6X,25I4)
0028      102   FORMAT(6X,12I10)
0029      103   FORMAT('1')
0030      104   FORMAT(' ',I5,25I4)
0031      105   FORMAT(' ',I5,12G10.3)
0032            END
```

```
0001            SUBROUTINE SCSMX(IRC,ICC,ICOL,A,MR,MC,NU)
0002      C     GAUSS ELIMINATION ALGORITHM
0003            REAL*8 A
0004            INTEGER*4 PIVROW,PIVCOL,OPROW
0005            COMMON /SINTG1/ PIVROW,PIVCOL,OPROW
0006            COMMON /SINTG2/ IY
0007            COMMON /SINTG3/ LKJ
0008            DIMENSION IRC(MR),ICC(MR),ICOL(MR,MC),A(MR,MC)
0009            PIVROW = LKJ
0010            PIVCOL = LKJ
0011            IY = 1
0012      C
0013      C     END GAUSS ELIMINATION
0014      C
0015            RETURN
0016            END
```

```
0001            SUBROUTINE SDSMX(IRC,ICC,ICOL,A,MR,MC,NU)
0002      C
0003      C     GAUSS JORDAN - PARTIAL PIVOTING ALGORITHM
0004            REAL*8 A,AA,ATEST
0005            INTEGER*4 PIVROW,PIVCOL,OPROW
0006            COMMON /SINTG1/ PIVROW,PIVCOL,OPROW
0007            COMMON /SINTG2/ IY
0008            COMMON /SINTG3/ LKJ
0009            DIMENSION IRC(MR),ICC(MR),ICOL(MR,MC),A(MR,MC)
0010            PIVCOL = LKJ
0011            ATEST = 0.D+00
0012            DO 712  I=1,NU
0013            IC = ICC(I)
0014            IF(IC.LE.0) GOTO 712
0015            II = ICOL(I,1)
0016            IF(II.GT.PIVCOL) GOTO 712
0017            AA = DABS(A(I,1))
0018            IF(AA.LE.ATEST) GOTO 712
0019            ATEST = AA
0020            PIVROW = I
0021      712   CONTINUE
0022            IY = 1
0023      C
0024      C     END GAUSS JORDAN-PARTIAL PIVOTING
0025            RETURN
0026            END
```

```
0001            SUBROUTINE SESMX(IRC,ICC,ICOL,A,MR,MC,NU)
0002     C
0003     C      GAUSS JORDAN FULL PIVOTING ALGORITHM
0004            REAL*8 A,AA,ATEST
0005            INTEGER*4 PIVROW,PIVCOL,OPROW
0006            COMMON /SINTG1/ PIVROW,PIVCOL,OPROW
0007            COMMON /SINTG2/ IY
0008            DIMENSION IRC(MR),ICC(MR),ICOL(MR,MC),A(MR,MC)
0009            ATEST = 0.D+00
0010            DO 710  I=1,NU
0011            IC = ICC(I)
0012            IF(IC.LE.0) GOTO 710
0013            DO 711 J=1,IC
0014            II = ICOL(I,J)
0015            IR = IRC(II)
0016            IF(IR.LE.0) GOTO 711
0017            AA = DABS(A(I,J))
0018            IF(AA.LE.ATEST) GOTO 711
0019            ATEST = AA
0020            PIVROW = I
0021            PIVCOL = II
0022            IY = J
0023     711    CONTINUE
0024     710    CONTINUE
0025     C
0026     C      END GAUSS JORDAN FULL PIVOTING
0027     C
0028            RETURN
0029            END
```

```
0001            SUBROUTINE SFSMX(IRC,ICC,ICOL,A,MR,MC,NU)
0002     C
0003     C      MINIMUM ROW - MINIMUM COLUMN ALGORITHM
0004     C
0005            INTEGER*4 PIVROW,PIVCOL,OPROW
0006            REAL*8 A
0007            COMMON /SINTG1/ PIVROW,PIVCOL,OPROW
0008            COMMON /SINTG2/ IY
0009            DIMENSION IRC(MR),ICC(MR),ICOL(MR,MC),A(MR,MC)
0010     C
0011     C      SELECT FIRST MIN ROW THEN FIRST MIN COL
0012     C      SELECT ROW WITH MINIMUM ENTRIES
0013            IK = 100000
0014            DO 700  I=1,NU
0015            IC = ICC(I)
0016            IF(IC.GE.IK.OR.IC.LE.0) GOTO 700
0017            PIVROW = I
0018            IK = IC
0019     700    CONTINUE
0020     C
0021     C      SELECT SMALLEST AVIABLE COLUMN FROM PIVROW
0022            IK = 100000
0023            IC = ICC(PIVROW)
0024            DO 701  I=1,IC
0025            II = ICOL(PIVROW,I)
0026            IR = IRC(II)
0027            IF(IR.GE.IK.OR.IR.LE.0) GOTO 701
0028            PIVCOL = II
0029            IK = IR
0030            IY = I
0031     701    CONTINUE
0032     C
0033     C      END FIRST MIN ROW THEN FIRST MIN COL
0034            RETURN
0035            END
```

```
0001            SUBROUTINE SGSMX(IRC,ICC,ICOL,A,MR,MC,NU)
0002    C
0003    C       MINIMUM COLUMN - MINIMUM ROW ALGORITHM
0004    C       SELECT FIRST MIN COL THEN FIRST MIN ROW
0005    C
0006            INTEGER*4 PIVROW,PIVCOL,OPROW
0007            REAL*8 A
0008            COMMON /SINTG1/ PIVROW,PIVCOL,OPROW
0009            COMMON /SINTG2/ IY
0010            DIMENSION IRC(MR),ICC(MR),ICOL(MR,MC),A(MR,MC)
0011            IK = 100000
0012            DO 706  I=1,NU
0013            IR = IRC(I)
0014            IF(IR.GE.IK.OR.IR.LE.0) GOTO 706
0015            PIVCOL = I
0016            IK = IR
0017    706     CONTINUE
0018            IK = 100000
0019            DO 707  I=1,NU
0020            IC = ICC(I)
0021            IF(IC.LE.0) GOTO 707
0022            DO 708  J=1,IC
0023            IF(ICOL(I,J).LT.PIVCOL) GOTO 708
0024            IF(ICOL(I,J).GT.PIVCOL.OR.IC.GE.IK) GOTO 707
0025            IK = IC
0026            PIVROW = I
0027            IY = J
0028    708     CONTINUE
0029    707     CONTINUE
0030    C
0031    C       END SELECT FIRST MIN COL THEN FIRST MIN ROW
0032    C
0033            RETURN
0034            END
```

```
0001            SUBROUTINE SHSMX(IRC,ICC,ICOL,A,MR,MC,NU)
0002      C     MAXIMUM COLUMN - MINIMUM ROW ALGORITHM
0003      C     SELECT FIRST MAX COL THEN FIRST MIN ROW
0004            INTEGER*4 PIVROW,PIVCOL,OPROW
0005            REAL*8 A
0006            COMMON /SINTG1/ PIVROW,PIVCOL,OPROW
0007            COMMON /SINTG2/ IY
0008            DIMENSION IRC(MR),ICC(MR),ICOL(MR,MC),A(MR,MC)
0009            IK = -1
0010            DO 703  I=1,NU
0011            IR = IRC(I)
0012            IF(IR.LE.IK.OR.IR.LE.0) GOTO 703
0013            PIVCOL = I
0014            IK = IR
0015      703   CONTINUE
0016            IK = 100000
0017            DO 704  I=1,NU
0018            IC = ICC(I)
0019            IF(IC.LE.0) GOTO 704
0020            DO 705  J=1,IC
0021            IF(ICOL(I,J).LT.PIVCOL) GOTO 705
0022            IF(ICOL(I,J).GT.PIVCOL.OR.IC.GE.IK) GOTO 704
0023            IK = IC
0024            PIVROW = I
0025            IY = J
0026      705   CONTINUE
0027      704   CONTINUE
0028      C
0029      C     END SELECT FIRST MAX COL THEN FIRST MIN ROW
0030      C
0031            RETURN
0032            END
```

```
0001            SUBROUTINE SISMX(IRC,ICC,ICOL,A,MR,MC,NU)
0002      C     MINIMUM ROW ENTRIES TIMES COLUMN ENTRIES ALGORITHM
0003      C     SELECT FIRST MIN (ROW.COL)
0004            INTEGER*4 PIVROW,PIVCOL,OPROW
0005            REAL*8 A
0006            COMMON /SINTG1/ PIVROW,PIVCOL,OPROW
0007            COMMON /SINTG2/ IY
0008            DIMENSION IRC(MR),ICC(MR),ICOL(MR,MC),A(MR,MC)
0009            IK = 100000
0010            DO 709  I=1,NU
0011            IC = ICC(I)
0012            IF(IC.LE.0) GOTO 709
0013            DO 702  J=1,IC
0014            II = ICOL(I,J)
0015            IR = IRC(II)
0016            IF(IR.LE.0) GOTO 702
0017            III = IC*IR
0018            IF(III.GE.IK) GOTO 702
0019            PIVROW = I
0020            PIVCOL = II
0021            IK = III
0022            IY = J
0023      702   CONTINUE
0024      709   CONTINUE
0025      C
0026      C     END SELECT FIRST MIN (ROW.COL)
0027            RETURN
0028            END
```

```
0001    C     ************************************************************
0002    C     TEST DU PROGRAMME DE RESOLUTION D'UN SYSTEME LINEAIRE DONT LA
0003    C     MATRICE EST CREUSE.
0004    C     ************************************************************
0005    C
0006    C     AUTEUR     : E. VERGISON (SOLVAY COMPUTER CENTER)
0007    C                  NOVEMBRE 1983.
0008    C
0009          REAL *8 A,AT,P
0010          DIMENSION A(50,50),P(50),AT(50,50)
0011          DIMENSION IRC(50),ICC(50),ICOL(50,50),IXA(50)
0012          EQUIVALENCE (N,NU)
0013          DATA LEC/5/,IMP/6/
0014          MR = 50
0015          MC = 50
0016          NU = 9
0017          ZTEST = 1.E-10
0018    C
0019    C     BOUCLE PRINCIPALE PARCOURANT LES SEPT ALGORITHMES DE RESOLUTION
0020    C     DE SYSTEMES LINEAIRES.
0021    C
0022          DO 10 NPIV = 1,7
0023    C
0024    C     INITIALISATIONS - MISES A ZERO.
0025    C
0026           DO 20 II = 1,NU
0027            IXA(II) = 0
0028            P(II) = 0.D+00
0029    20     CONTINUE
0030           DO 30  J = 1,NU
0031           DO 30  I = 1,NU
0032            A(I,J) = 0.D+00
0033            AT(I,J) = 0.D+00
0034            ICOL(I,J) = 0
0035    30     CONTINUE
0036    C
0037    C     DEFINITION DU SECOND MEMBRE DU SYSTEME LINEAIRE.
0038    C
0039           P(1) = -2.D+00
0040           P(2) = -1.D+00
0041           P(3) =  4.D+00
0042           P(4) =  3.D+00
```

```
0043            P(6) =  7.D+00
0044            P(7) = 16.D+00
0045            P(8) = 11.D+00
0046            P(9) = 22.D+00
0047    C
0048    C       MATRICE EXEMPLATIVE.
0049    C
0050            DO 40 I = 1,9
0051             AT(I,I) = 4.D+00
0052    40      CONTINUE
0053            DO 50 I = 1,6
0054             AT(I,I+3) = -1.D+00
0055             AT(I+3,I) = -1.D+00
0056    50      CONTINUE
0057            AT(1,2) = -1.D+00
0058            AT(2,3) = -1.D+00
0059            AT(4,5) = -1.D+00
0060            AT(5,6) = -1.D+00
0061            AT(7,8) = -1.D+00
0062            AT(8,9) = -1.D+00
0063            AT(2,1) = -1.D+00
0064            AT(3,2) = -1.D+00
0065            AT(5,4) = -1.D+00
0066            AT(6,5) = -1.D+00
0067            AT(8,7) = -1.D+00
0068            AT(9,8) = -1.D+00
0069    C
0070    C       ALGORITHME DE COMPACTAGE.
0071    C
0072            DO 60  J = 1,NU
0073            DO 60  I = 1,NU
0074             IF(AT(I,J) .EQ.0.D+00) GO TO 60
0075             IXA(I) = IXA(I) + 1
0076             ICOL(I,IXA(I)) = J
0077             A(I,IXA(I)) = AT(I,J)
0078    60      CONTINUE
0079    C
0080    C
0081    C       ALGORITHME DE RESOLUTION DE SYSTEMES LINEAIRES CREUX.
0082    C
0083             CALL SOSMX(NU,MR,MC,A,P,ICOL,ZTEST,ICC,IRC,JELE,JELIM,JCOL,NPIV,
0084           1 TIME)
```

```
0085    C
0086    C       IMPRESSION DES RESULTATS.
0087    C
0088             WRITE(IMP,2000) JELE,JELIM,JCOL,TIME
0089             WRITE(IMP,2100)((I,A(I,1)),I=1,NU)
0090    10      CONTINUE
0091    C
0092    C       FORMATS.
0093    C
0094    2000     FORMAT(1H ,5X,'JELE = ',I5,5X,'JELIM = ',I5,5X,'JCOL = ',I5, 5X,'
0095           1TIME = ',F8.2,//)
0096    2100     FORMAT(1H ,2X,I3,2X,F6.2,1X,';',2X,I3,2X,F6.2,1X,';',
0097           1         2X,I3,2X,F6.2,1X,';',2X,I3,2X,F6.2,1X,';')
0098             CALL EXIT
0099             END
```

# ANNEXE 3

Résolution d'un problème non linéaire par la méthode multigrille

- Programme de calcul

- Résultats

```
0001   C*****************************************************************
0002   C
0003   C         MAIN PROGRAM FOR DEMONSTRATION OF MGOOD, AN FMG POISSON
0004   C         SOLVER EXTENDED TO A NONLINEAR ELLIPTIC EQUATION OF THE
0005   C         FORM
0006   C           DIFFERENTIAL EQUATION :-DELTA U(X,Y) + C(X,Y,U) = F(X,Y)
0007   C           BOUNDARY CONDITION    :                   U(X,Y) = G(X,Y)
0008   C         BY NEWTON TECHNIQUE.
0009   C
0010   C         BASIC PAPERS BY FOERSTER, H. AND WITSCH, K. IN
0011   C           MULTIGRIDS METHODS - PROCEEDINGS, KOLN-PORZ, 1981
0012   C           EDITED BY W. HACKBUSH AND U. TROTTENBERG,
0013   C           SPRINGER VERLAG - 1982
0014   C         AND K. STUBEN AND U. TROTTENBERG (SAME REFERENCE).
0015   C
0016   C         THE EXTENSION WAS MADE BY M.-A. LAMBOTTE (FACULTES
0017   C         UNIVERSITAIRES DE NAMUR) AND E. VERGISON (SOLVAY S.A.)
0018   C
0019   C         THE EXAMPLE GIVEN HERE COMES FROM R. TEMAM IN
0020   C           ANALYSE NUMERIQUE,
0021   C           PRESSE UNIVERSITAIRE DE FRANCE - 1970 - P.110
0022   C         THE VALUE TAKEN FOR PARAMETER LAMDA IS 100.
0023   C
0024   C*****************************************************************
0025   C
0026         DOUBLE PRECISION F, G, C, CPRIM, DIFMX, DEFMX, DIF, DEF
0027         INTEGER IDIM, IER, M, NY1, NY2, NCYCLE, NFMG, IGAM, INITF
0028         INTEGER ITERMAX, NMID, LEC, IMP
0029         DOUBLE PRECISION W(18000)
0030         EXTERNAL F, G, C, CPRIM
0031         DATA LEC /5/, IMP /6/
0032   C
0033         IDIM = 18000
0034         NY1 = 2
0035         NY2 = 1
0036   C
0037   C     ENTER THE DATA
0038   C
0039   C     M      : NUMBER OF GRIDS (0< M <11)
0040   C     NCYCLE : NUMBER OF MULTIGRIDS ITERATIONS (NCYCLE > 0)
0041   C     NFMG   : .EQ. 0 : NCYCLE MULTIGRIDS ITERATIONS ARE PERFORMED
0042   C                .NE. 0 : FMG-VERSION IS PERFORMED
```

```
0043    C       IGAM    : TYPE OF CYCLING (IGAM > 0)
0044    C                       IGAM .EQ. 1 : V-CYCLE
0045    C                       IGAM .EQ. 2 : W-CYCLE
0046    C       ITERMAX: MAXIMUM NUMBER OF ITERATION IN THE NEWTON ITERATION
0047    C                PROCEDURE.
0048    C
0049            READ(LEC,*) M, NCYCLE, NFMG, IGAM, ITERMAX
0050    C
0051            CALL MGOOD(M,NY1,NY2,NFMG,NCYCLE,IGAM,F,G,C,CPRIM,W,IDIM,INITF,
0052           1           IER,ITERMAX)
0053    C
0054            IF(IER.EQ.0) GOTO 10
0055            WRITE(IMP,2000) IER,IDIM
0056            CALL EXIT
0057    10      CONTINUE
0058    C
0059    2000    FORMAT(16H *** ERROR,IER=, I3, 7H, IDIM=, I6, 4H ***)
0060            CALL EXIT
0061            END
```

3-MAY-84 16:48

```
0001    C
0002    C-----------------------------------------------------------------------
0003    C
0004          DOUBLE PRECISION FUNCTION F(X,Y)
0005          DOUBLE PRECISION X, Y, DSIN
0006          F = -200.D+00*X*(X-1) - 200.D+00*Y*(Y-1)
0007         1    + 1.D+04*X*(X-1.D+00)*Y*(Y-1.D+00)
0008         2    + 1.D+06*(X*(X-1)*Y*(Y-1))**3
0009          RETURN
0010          END
```

```
0001    C
0002    C-----------------------------------------------------------------------
0003    C
0004          DOUBLE PRECISION FUNCTION G(X,Y)
0005          DOUBLE PRECISION X, Y, DSIN
0006          G = 100.D+00*X*(X-1)*Y*(Y-1)
0007          END
```

```
0001    C
0002    C-----------------------------------------------------------------------
0003    C
0004          DOUBLE PRECISION FUNCTION C(X,Y,U)
0005          DOUBLE PRECISION X, Y, U
0006          C = U*(100.D+00 + U*U)
0007          RETURN
0008          END
```

```
0001    C
0002    C-----------------------------------------------------------------------
0003    C
0004          DOUBLE PRECISION FUNCTION CPRIM(X,Y,U)
0005          DOUBLE PRECISION X, Y, U
0006    C
0007    C     PARTIAL DERIVATIVE WITH RESPECT TO U OF FUNCTION C(X,Y,U)
0008    C
0009          CPRIM = 100.D+00 + 3.D+00*U*U
0010          RETURN
0011          END
```

```
0001    C
0002    C*********************************************************************
0003    C
0004    C                    M G O O D    VERSION 20/02/84
0005    C
0006    C*********************************************************************
0007    C
0008          SUBROUTINE MGOOD(M, NY1, NY2, NFMG, NCYCLE, IGAM, F, G, C, CPRIM,
0009         1                 W, IDIM, INITF, IER, ITERMAX)
0010    C
0011    C     MULTIGRID MODULE FOR THE FAST SOLUTION OF A NONLINEAR ELLIPTIC
0012    C     PARTIAL DIFFERENTIAL EQUATION WITH DIRICHLET BOUNDARY CONDITIONS
0013    C     ON THE UNIT SQUARE :
0014    C
0015    C       DIFFERENTIAL EQUATION : -DELTA U(X,Y) + C(X,Y,U(X,Y)) = F(X,Y)
0016    C       BOUNDARY CONDITION    :                         U(X,Y) = G(X,Y)
0017    C
0018    C     CYCLE STRUCTURE :
0019    C
0020    C                COARSENING      : H, 2H, 4H, ...
0021    C                DIFFERENCE OPERATORS: USUAL 5-POINT STARS ON ALL GRIDS
0022    C                RELAXATION      : RED-BLACK
0023    C                FINE-TO-COARSE  : HALF INJECTION
0024    C                COARSE-TO-FINE  : BILINEAR INTERPOLATION
0025    C
0026    C     FULL MULTIGRID MODE:
0027    C
0028    C                FULL MULTIGRID INTERPOLATION USES GRID EQUATION
0029    C                (4-TH ORDER)
0030    C
0031    C     INPUT :
0032    C
0033    C        M          NUMBER OF GRIDS (0 < M < 11). FOR GIVEN M:
0034    C                   NUMBER OF POINTS ON THE FINEST GRID (INCLUDING BOUNDARY
0035    C                   POINTS) = 2**M + 1 IN BOTH DIRECTIONS
0036    C        NY1        NUMBER OF RELAXATIONS BEFORE COARSE-GRID CORRECTION
0037    C                   (NY1 > 0)
0038    C        NY2        NUMBER OF RELAXATIONS AFTER COARSE-GRID CORRECTION
0039    C                   (NY2 > 0)
0040    C        NCYCLE     NUMBER OF MULTIGRID ITERATIONS (NCYCLE > 0)
0041    C        NFMG       .EQ. 0 : NCYCLE MULTIGRID ITERATIONS ARE PERFORMED
0042    C                   .NE. 0 : FMG-VERSION IS PERFORMED PLUS NCYCLE-1
```

```
0043    C                                ADDITIONAL MG ITERATIONS AFTERWARDS
0044    C        IGAM          TYPE OF CYCLING (IGAM > 0).
0045    C                      E.G.: IGAM=1 FOR V-CYCLES, IGAM=2 FOR W-CYCLES
0046    C        W             DOUBLE PRECISION WORK ARRAY OF DIMENSION IDIM
0047    C        IDIM          DIMENSION OF W. APPROXIMATELY: IDIM > 2.8*4**M
0048    C        ITERMAX        MAXIMUM NUMBER OF ITERATIONS IN THE FIXED POINT
0049    C                       PROCEDURE.
0050    C
0051    C    EXTERNALS:
0052    C
0053    C        F             DOUBLE PRECISION FUNCTION F(X,Y), RIGHT HAND SIDE OF
0054    C                      THE DIFFERENTIAL EQUATION
0055    C        G             DOUBLE PRECISION FUNCTION G(X,Y), BOUNDARY VALUES
0056    C        C             DOUBLE PRECISION FUNCTION C(X,Y,U(X,Y)),
0057    C                      NON LINEAR HELMHOLTZ TERM SUPPOSED TO BE NON
0058    C                      NEGATIVE.
0059    C        CPRIM         DOUBLE PRECISION FUNCTION CPRIM(X,Y,U(X,Y)),
0060    C                      PARTIAL DERIVATIVE WITH RESPECT TO "U" OF
0061    C                      FUNCTION C(X,Y,U)
0062    C
0063    C    REMARK:           GRID #1 AND #M ARE THE FINEST AND COARSEST GRID USED,
0064    C                      RESPECTIVLY
0065    C
0066    C    OUTPUT:
0067    C
0068    C        IER           ERROR INDICATOR
0069    C                      = 0 NO ERRORS
0070    C                      = 1 INSUFFICIENT MEMORY, I.E. IDIM TOO SMALL.
0071    C                          IN THIS CASE IDIM IS USED AS OUTPUT PARAMETER
0072    C                          TO SHOW THE MINIMAL DIMENSION
0073    C                      = 2 M, NY1, NY2, NCYCLE OR IGAM WRONG
0074    C        IDIM          ONLY IN CASE IER=1 : MINIMAL LENGTH OF W
0075    C        INITF         CF. DESCRIPTION OF W
0076    C        W             W CONTAINS THE DISCRETE APPROXIMATION TO THE GIVEN
0077    C                      BOUNDARY VALUE PROBLEM ON THE FINEST GRID. THE GRID
0078    C                      VALUES ARE STORED ROWWISE FROM LEFT TO RIGHT AND
0079    C                      FROM BOTTOM TO TOP.
0080    C                      I.E. THE GRID VALUE CORRESPONDING TO THE GRID POINT
0081    C                      (XI,YJ) :
0082    C
0083    C                          XI = (I-1)*H, YJ = (J-1)*H (H=1/N, N=2**M)
0084    C
```

```
0085    C                     IS STORED AT
0086    C                           W((J-1)*(N+1)+I)     (0 < I,J < N+2)
0087    C
0088    C                     THE CORRESPONDING VALUES OF THE RIGHT HAND SIDE ARE
0089    C                     STORED IN THE SAME MANNER AT
0090    C
0091    C                           W((J-1)*(N+1)+I+INITF).
0092    C
0093    C                      THE CORRESPONDING VALUES OF THE HELMOLTZ TERM
0094    C                      IS STORED AT
0095    C                           W((J-1)*(N+1)+I+2*INITF)
0096    C
0097    C                     THE REMAINING STORAGE CONTAINS COARSE GRID VALUES.
0098    C
0099          INTEGER IDC, IDF, IDR, IDS, IT, ITM, INITF, L, L1, LEV, LIN, NP,
0100         *NY1, NY2, IGAM, IDIM, IER, NCYCLE, NFMG, M, N, ID(11), NPK(10)
0101          INTEGER ITER, ITERMAX, NMID
0102          DOUBLE PRECISION W(1), HK(10), DFLOAT, F, G, C, CPRIM
0103          EXTERNAL F, G, C, CPRIM
0104          DFLOAT(L1) = DBLE(FLOAT(L1))
0105          IER = 0
0106    C
0107          IF(M.LE.0 .OR. M.GE.11 .OR. NY1.LE.0 .OR. NY2.LE.0 .OR.
0108         *   NCYCLE.LE.0 .OR. IGAM.LE.0 ) IER=2
0109          IF(IER.GT.0) GOTO 50
0110    C
0111    C     DETERMINATION OF COARSER GRIDS
0112    C
0113          NP    = 2**M + 1
0114          N     = NP - 1
0115          NMID = (NP*NP + 1)/2
0116          HK(1) = 1.0D0/DFLOAT(NP-1)
0117          ID(1) = 1
0118          ID(2) = NP*NP + 1
0119          NPK(1) = NP
0120          DO 10 L = 2,M
0121            NPK(L) = (NPK(L-1) + 1)/2
0122            HK(L) = HK(L-1) + HK(L-1)
0123            ID(L+1) = ID(L) + NPK(L)**2
0124    10    CONTINUE
0125    C
0126    C           CHECK OF DIMENSIONS
```

```
0127    C
0128          INITF = ID(M+1)
0129          IF(3*INITF.LE.IDIM) GOTO 20
0130          IER = 1
0131          IDIM = 3*INITF
0132          GOTO 60
0133    20    LIN = 1
0134          IF(NFMG.NE.0) LIN = M
0135          DO 50 ITER = 1,ITERMAX
0136    C
0137    C           SET UP ALL GRID VALUES NEEDED
0138    C
0139          CALL INIT1(NP, HK(1), F, G, C, CPRIM,
0140         1                          W(1), W(INITF+1), W(2*INITF+1),ITER)
0141          CALL INIT2(M, LIN, NPK, ID ,W(1), W(INITF+1), W(2*INITF+1),INITF)
0142    C
0143    C            (FULL) MULTIGRID PROCEDURE
0144    C
0145          DO 40  L = 1,LIN
0146           LEV = LIN - L + 1
0147           ITM = 1
0148           IF(LEV.EQ.1) ITM=NCYCLE
0149           DO 30  IT = 1,ITM
0150           CALL MGI(LEV, M, NY1, NY2, IGAM, NPK, ID, W, W(INITF+1),
0151         1         W(2*INITF+1), INITF)
0152    30     CONTINUE
0153          IF(L.LT.LIN) THEN
0154            IDF = ID(LEV-1)
0155            IDC = ID(LEV)
0156            IDR = IDF + INITF
0157            IDS = IDR + INITF
0158            CALL INT4(NPK(LEV), NPK(LEV-1), W(IDC), W(IDF), W(IDR), W(IDS))
0159            CALL PUTZB(NPK(LEV),W(IDC))
0160          END IF
0161    40    CONTINUE
0162          DIF = DIFMX(W,NP,G)
0163          DEF = DEFMX(W,W(INITF+1),W(2*INITF+1),NP)
0164          WRITE(6,9100) N, DIF, DEF
0165          WRITE(6,9200) W(NMID)
0166    C
0167    9100   FORMAT(30H NUMBER OF INTERVALS          = , I4, /,
0168         1       30H MAXIMUM NORM OF THE ERROR  = , D12.4, /,
```

```
0169          2       30H MAXIMUM NORM OF THE DEFECT = , D12.4)
0170    9200  FORMAT(1HO,30X,'VALEUR CENTRALE DE LA SOLUTION = ',D12.5)
0171    C
0172    50    CONTINUE
0173    60    RETURN
0174          END
```

```
0001    C
0002    C-----------------------------------------------------------------------
0003    C
0004          SUBROUTINE INIT1(NP, H, F, G, C, CPRIM, U, FR, CR, ITER)
0005    C
0006    C         COMPUTES INITIAL VALUES ON THE FINEST GRID
0007    C
0008          INTEGER I, J, N, NP
0009          DOUBLE PRECISION FR(NP,NP), CR(NP,NP), U(NP,NP), DFLOAT
0010          DOUBLE PRECISION F, G, C, CPRIM
0011           DOUBLE PRECISION H, H2, X, Y
0012          DFLOAT(K) = DBLE(FLOAT(K))
0013    C
0014          H2 = H*H
0015          N = NP - 1
0016          IF(ITER.EQ.1) THEN
0017          DO 20  J = 2,N
0018            Y = DFLOAT(J-1)*H
0019            U(1,J) = G(0.0D0,Y)
0020            DO 10 I = 2,N
0021              X = DFLOAT(I-1)*H
0022              U(I,J) = 0.0D0
0023              FR(I,J) = H2*(F(X,Y) - C(X,Y,U(I,J))
0024         1                          + CPRIM(X,Y,U(I,J))*U(I,J))
0025               CR(I,J) = H2*CPRIM(X,Y,U(I,J))
0026    10      CONTINUE
0027            U(NP,J) = G(1.0D0,Y)
0028    20      CONTINUE
0029          DO 30  I = 1,NP
0030            X = DFLOAT(I-1)*H
0031            U(I,1) = G(X,0.0D0)
0032            U(I,NP) = G(X,1.0D0)
0033    30      CONTINUE
0034          ELSE
0035          DO 40 J = 2,N
0036            Y = DFLOAT(J-1)*H
0037            DO 50 I = 2,N
0038              X = DFLOAT(I-1)*H
0039              FR(I,J) = H2*(F(X,Y) - C(X,Y,U(I,J))
0040         1                          + CPRIM(X,Y,U(I,J))*U(I,J))
0041               CR(I,J) = H2*CPRIM(X,Y,U(I,J))
0042    50      CONTINUE
```

```
0043    40      CONTINUE
0044            END IF
0045            RETURN
0046            END
```

```
0001    C
0002    C-----------------------------------------------------------------------
0003    C
0004          SUBROUTINE INIT2(M, LIN, NPK, ID, U, FR, CR, IDIM)
0005    C
0006    C         COMPUTES INITIAL VALUES ON THE COARSER GRIDS
0007    C
0008          INTEGER IDIM, LIN, M, IDC, IDF, L, LH, ID(11), NPK(10)
0009          DOUBLE PRECISION FR(IDIM), CR(IDIM), U(IDIM)
0010    C
0011          IF(LIN.EQ.1) GOTO 20
0012    C
0013    C         TRANSFER OF F AND U TO COARSER GRIDS
0014    C
0015          DO 10  L = 2,LIN
0016            IDC = ID(L)
0017            IDF = ID(L-1)
0018            CALL TRANS(NPK(L), NPK(L-1), U(IDC), FR(IDC), CR(IDC),
0019         1                               U(IDF), FR(IDF), CR(IDF))
0020    10    CONTINUE
0021    C
0022    C         PUT ZERO TO BOUNDARY VALUES FOR MULTIGRID CORRECTONS
0023    C
0024    20    IF(LIN.EQ.M) GOTO 40
0025          LH = LIN + 1
0026          DO 30  L = LH,M
0027            IDC = ID(L)
0028            CALL PUTZB(NPK(L), U(IDC))
0029    30    CONTINUE
0030    40    RETURN
0031          END
```

```
0001    C
0002    C-----------------------------------------------------------------
0003    C
0004          SUBROUTINE TRANS(NPC, NPF, UC, FC, CC, UF, FF, CF)
0005    C
0006    C        TRANSFER OF F, C AND U FROM GRID NPF TO NPC
0007    C
0008          INTEGER NC, NPC, NPF, I, IF, J, JF
0009          DOUBLE PRECISION FC(NPC,NPC), CC(NPC,NPC), UC(NPC,NPC),
0010         1                 FF(NPF,NPF), CF(NPF,NPF), UF(NPF,NPF)
0011    C
0012          NC = NPC - 1
0013    C
0014          DO 20  J = 2,NC
0015            JF = J + J - 1
0016            UC(1,J) = UF(1,JF)
0017            DO 10  I=2,NC
0018              IF = I + I - 1
0019              UC(I,J) = UF(IF,JF)
0020              FC(I,J) = 4.D+00*FF(IF,JF)
0021               CC(I,J) = 4.D+00*CF(IF,JF)
0022    10      CONTINUE
0023            UC(NPC,J) = UF(NPF,JF)
0024    20      CONTINUE
0025            DO 30  I = 1,NPC
0026              IF = I + I - 1
0027              UC(I,1) = UF(IF,1)
0028              UC(I,NPC) = UF(IF,NPF)
0029    30      CONTINUE
0030            RETURN
0031            END
```

```
0001    C
0002    C-----------------------------------------------------------------------
0003    C
0004          SUBROUTINE PUTZB(NPC, UC)
0005    C
0006    C         PUTS ZERO TO BOUNDARY OF COARSER GRIDS
0007    C
0008          INTEGER NC, NPC, I, J
0009          DOUBLE PRECISION UC(NPC,NPC)
0010    C
0011          NC = NPC-1
0012          DO 10  J = 2,NC
0013            UC(1,J) = 0.0D0
0014            UC(NPC,J) = 0.0D0
0015    10    CONTINUE
0016          DO 20  I = 1,NPC
0017            UC(I,1) = 0.0D0
0018            UC(I,NPC) = 0.0D0
0019    20    CONTINUE
0020          RETURN
0021          END
```

```
0001    C-----------------------------------------------------------------------
0002    C
0003          SUBROUTINE MGI( LEV, M, NY1, NY2, IGAM, NPK, ID, U, FR, CR, IDIM)
0004    C
0005    C        ONE MULTIGRID ITERATION STEP (ON ACTUAL FINEST GRID LEV)
0006    C
0007          INTEGER IDIM, LEV, M, NY1,NY2, ID(11), NPK(10), IDC, IDF, IZ,
0008         1        K, IGAM, ICGAM(10)
0009          DOUBLE PRECISION FR(IDIM), CR(IDIM), U(IDIM)
0010    C
0011          IZ = 1
0012          DO 10  K = LEV,M
0013            ICGAM(K) = 0
0014    10    CONTINUE
0015          K = LEV
0016          IF(K.EQ.M) GOTO 30
0017    20    IDF = ID(K)
0018    C
0019    C         RELAXATIONS BEFORE CGC
0020    C
0021             IZ = 1
0022             IF(K.GT.LEV.AND.ICGAM(K).EQ.0) IZ = 0
0023             CALL RELAX(NY1+NY1, IZ, NPK(K),U(IDF),FR(IDF),CR(IDF),FR(1))
0024             ICGAM(K) = ICGAM(K) + 1
0025    C
0026    C         RESIDUAL TRANFER TO NEXT COARSER GRID
0027    C
0028             IDC = ID(K+1)
0029             CALL RESTR(NPK(K+1), NPK(K), FR(IDC), U(IDF), FR(IDF), CR(IDF))
0030             K = K + 1
0031             IF(K.LT.M) GOTO 20
0032          IZ = 0
0033    30    IDC = ID(M)
0034    C
0035    C        EXACT SOLUTION ON COARSEST GRID
0036    C
0037          CALL RELAX(1, IZ, NPK(M), U(IDC), FR(IDC), CR(IDC), FR(1))
0038          IF(K.EQ.LEV) GOTO 50
0039    40      K = K - 1
0040            IDF = ID(K)
0041            IDC = ID(K+1)
0042    C
```

```
0043    C          LINEAR INTERPOLATION TO NEXT FINER GRID
0044    C
0045              CALL INT2A(NPK(K+1), NPK(K), U(IDC), U(IDF))
0046    C
0047    C          RELAXATION AFTER CGC
0048    C
0049              CALL RELAX(NY2+NY2, 1, NPK(K), U(IDF), FR(IDF), CR(IDF), FR(1))
0050             IF(K.EQ.LEV) GOTO 50
0051             IF(ICGAM(K).LT.IGAM) GOTO 20
0052             ICGAM(K) = 0
0053             GOTO 40
0054    50       RETURN
0055             END
```

```
0043                DIAG = 1.D+00/(4.D+00 + CF(I,J))
0044                UF(I,J) = DIAG*(FF(I,J) + W(I) + W(I+1))
0045    50          CONTINUE
0046                IS = 5 - IS
0047    60        CONTINUE
0048    70      CONTINUE
0049    C
0050            RETURN
0051            END
```

```
0001    C
0002    C-----------------------------------------------------------------------
0003    C
0004          SUBROUTINE RESTR(NPC, NPF, FC, UF, FF, CF)
0005    C
0006    C       COMPUTATION OF THE DEFECT AND FINE-TO-COARSE TRANSFER
0007    C       (HALF-INJECTION)
0008    C
0009          INTEGER NC, NPC, NPF, I, IF, J, JF
0010          DOUBLE PRECISION FC(NPC,NPC),FF(NPF,NPF),CF(NPF,NPF),UF(NPF,NPF)
0011          DOUBLE PRECISION H
0012    C
0013          NC = NPC - 1
0014          DO 20  J = 2,NC
0015            JF = J + J - 1
0016            DO 10  I = 2,NC
0017              IF = I + I - 1
0018              H = FF(IF,JF) - (4.D+00 + CF(IF,JF))*UF(IF,JF)
0019         1         + UF(IF,JF-1) + UF(IF-1,JF) + UF(IF+1,JF) + UF(IF,JF+1)
0020              FC(I,J) = H + H
0021    10        CONTINUE
0022    20      CONTINUE
0023          RETURN
0024          END
```

```
0001    C
0002    C-----------------------------------------------------------------------
0003    C
0004          SUBROUTINE INT2A(NPC, NPF, UC, UF)
0005    C
0006    C        COARSE-TO-FINE TRANSFER (BILINEAR INTERPOLATION) AND CORRECTION
0007    C
0008          INTEGER NC, NPC, NPF, I, IF, J, JF
0009          DOUBLE PRECISION UC(NPC,NPC), UF(NPF,NPF)
0010    C
0011          NC = NPC - 1
0012          DO 20  J = 2,NC
0013            JF = J + J - 1
0014            DO 10  I = 1,NC
0015              IF = I + I
0016              UF(IF,JF) = UF(IF,JF) + 0.5D0*(UC(I,J)+UC(I+1,J))
0017    10        CONTINUE
0018    20      CONTINUE
0019          DO 40  J = 1,NC
0020            JF = J + J
0021            DO 30  I = 2,NC
0022              IF = I + I - 1
0023              UF(IF,JF) = UF(IF,JF) + 0.5D0*(UC(I,J) + UC(I,J+1))
0024    30        CONTINUE
0025    40      CONTINUE
0026          RETURN
0027          END
```

```
0043    C
0044    C       COMPUTATION OF THE ODD POINTS BY ONE HALF (ODD) RELAXATION STEP
0045    C
0046            CALL RELAX(-1, 1, NPF, UF, FRF, CRF, FRF)
0047            RETURN
0048            END
```

```
0001    C
0002    C-----------------------------------------------------------------------
0003    C
0004          DOUBLE PRECISION FUNCTION DIFMX(UC, NP, SOL)
0005    C
0006    C      COMPUTE THE MAXIMUM NORM OF THE DIFFERENCE BETWEEN
0007    C      SOL (=SOLUTION OF THE BVP) AND THE VALUES IN UC
0008    C
0009          INTEGER NP, I, J
0010          DOUBLE PRECISION UC(NP,NP), SOL, DFLOAT, H, X, Y
0011          DFLOAT(K) = DBLE(FLOAT(K))
0012    C
0013          H = 1.0D0/DFLOAT(NP-1)
0014          DIFMX = 0.0D0
0015          DO 20  J = 1,NP
0016            Y = DFLOAT(J-1)*H
0017            DO 10  I = 1,NP
0018              X = DFLOAT(I-1)*H
0019              DIFMX = DMAX1(DIFMX,DABS(SOL(X,Y) - UC(I,J)))
0020    10      CONTINUE
0021    20    CONTINUE
0022          RETURN
0023          END
```

```
0001    C
0002    C-----------------------------------------------------------------------
0003    C
0004          DOUBLE PRECISION FUNCTION DEFMX(UC, FC, CC, NP)
0005    C
0006    C        COMPUTES THE MAXIMUM NORM OF THE DEFECT
0007    C
0008          INTEGER N, NP, I, J
0009          DOUBLE PRECISION UC(NP,NP), FC(NP,NP), CC(NP,NP), D, HSQR
0010    C
0011          N = NP - 1
0012          HSQR = DBLE(FLOAT(N*N))
0013          DEFMX = 0.0D0
0014          DO 20  J = 2,N
0015            DO 10  I = 2,N
0016              D = (FC(I,J) - (4.D+00+CC(I,J))*UC(I,J)+UC(I-1,J)+UC(I,J-1)
0017         1           + UC(I,J+1) + UC(I+1,J))*HSQR
0018              DEFMX = DMAX1(DEFMX,DABS(D))
0019    10        CONTINUE
0020    20    CONTINUE
0021          RETURN
0022          END
```

[ 9] G. W. STEWART : Introduction to Matrix Computations. Academic Press, 1973.

[10] K. STUBEN and U. TROTTENBERG : Multigrid Methods: Fundamental Algorithmes, Model Problem Analysis and Applications. Proceedings, Köln-Porz, 1981. Lecture Notes in Mathematics, 960. Springer-Verlag, Berlin, 1982.

[11] K. STUBEN : Méthodes Multigrilles. Cours Gamni, Ecole Centrale de Lyon, septembre 1983.

[12] R. TEMAM : Analyse Numérique. Presses Universitaires de France, 1970.

[13] U. TROTTENBERG : Méthodes Multigrilles. Cours Gamni, Ecole Centrale de Lyon, septembre 1983.

[14] D. WEXLER : Equations aux dérivées partielles, cours donné aux FNDP Namur, 1983.

[15] O. ZIENKIEWICZ : The finite element method in Engineering Science. Mac Graw Hill, 1971.